

Implementación de arquitectura en AWS para publicación de sitios Web

PRESENTADO POR:

Jorge Orlando Ramírez Agudelo

Juan Pablo Cadavid Monsalve

Jorge Leonardo Álvarez

DOCENTE:

Juan Pablo Berrio López

Seminario AWS

Mayo de 2025

Dedicatoria

Para Juan Pablo Berrio López:

Por su excelente metodología de enseñanza, por su admirable vocación como docente y por su metodología de enseñanza, logró facilitarnos este proceso de aprendizaje e inspirarnos para dar lo mejor de nosotros en este proceso.

Gracias por compartir su conocimiento, ha sido una excelente experiencia.

Agradecimientos

Queremos expresar nuestra más profunda gratitud a nuestras familias, quienes han sido un pilar fundamental durante todo este proceso.

Gracias por su paciencia, apoyo y palabras de aliento en los momentos de mayor esfuerzo. Su comprensión y confianza nos motivaron a seguir adelante y dar lo mejor de nosotros.

Este logro también es de ustedes.

Tabla de Contenidos

Contenido

Dedicatoria.....	2
Agradecimientos	3
Tabla de Contenidos	4
Marco conceptual.....	6
Marco contextual	6
Desarrollo e Implementación del aprendizaje.....	7
Diagrama de arquitectura.....	7
Descripción de la arquitectura	8
Tipo de instancias usadas (Linux: Amazon Linux. Windows: Windows Server).	8
Justificación de las configuraciones de red	8
Configuraciones realizadas	8
Pasos para crear las instancias EC2.	8
Detalles de los Grupos de Seguridad (puertos abiertos: RDP, SSH, HTTP).....	9
1. Windows	9
2. Linux	9
Asignación de IPs públicas y privadas.....	10
1. Windows	10
1. Windows	11
2. Linux	13
Configuración del servidor web.....	15
Pasos seguidos para instalar IIS en Windows Server.	15
Pasos para instalar Apache en Linux.	17
Pruebas de conectividad.....	19
Ping desde Windows a Linux	19
Ping desde Linux a Windows	20
PRUEBAS DOCKER	22
Que es Docker:.....	22
Qué son las Máquinas Virtuales (VMs):.....	22
Diferencias clave entre Docker y Máquinas virtuales	22
Ventajas del Docker frente a las Máquinas virtuales.....	23
Procedimiento de un Docker.....	23
Pasos Para Detener el Docker	32
Conclusiones	35
Referencias.....	37

Resumen

El siguiente proyecto tuvo como objetivo principal realizar el diseño e implementación de una red en la plataforma Amazon Web Services (AWS), mediante la creación en una Virtual Private Cloud (VPC), subredes públicas y privadas, direcciones IP dinámicas y grupos de seguridad, y progresivamente instancias EC2 totalmente personalizadas.

La red fue conformada por dos instancias EC2 (tipo t2.micro, gratuita de AWS) que opera bajo el sistema operativo Amazon Linux 2023 y Windows Server 2016, simulando un entorno mixto que permite realizar pruebas de interoperabilidad, conectividad remota (mediante SSH y RDP).

La arquitectura diseñada se encuentra dentro del nivel gratuito de AWS, lo cual facilitó su implementación evitando algún costo de AWS, de uso netamente para fines académicos y de experimentación. Estas instancias fueron configuradas para ofrecer servicios web básicos, permitiendo validar la comunicación y funcionalidad dentro de la red virtual (VPC) creada.

Adicionalmente, se usaron metodologías de contenedores con la librería de Docker; para poder alojar y ejecutar esta aplicación en un entorno controlado, poder tener una gestión de estados de recursos y la escalabilidad del sistema. Esta actividad nos permitió realizar múltiples pruebas relacionadas con la accesibilidad y rendimiento de los servicios con pruebas de estrés y consumo de recursos.

Los resultados obtenidos dieron a entender el correcto funcionamiento de una infraestructura, así como la confianza de integrar servicios en la nube con tecnologías de virtualización para el desarrollo de soluciones modernas y eficientes.

Marco conceptual

Implementación de arquitectura en AWS para publicación de sitios Web

Este trabajo se construyó con la implementación de tecnologías de computación en la nube (esencialmente AWS), para aumentar la eficiencia en implementaciones, gestión y mejora de infraestructuras.

Entre los servicios utilizados en este proyecto encontramos:

- 1) EC2 (Elastic Compute Cloud): Creador de máquinas virtuales o instancias en nube
- 2) VPC (Virtual Private Cloud): Red virtual privada para definir subredes, dirección IP, reglas y mecanismos de seguridad.
- 3) Grupo de seguridad: A través de los controles de firewall y las claves de acceso se permite poder proteger la integridad de la conexión.
- 4) Docker: Manejo de contenedores para poder generar entornos independientes de servicios para que sea más eficiente y ágil el consumo de recursos, que nos permite mejorar la escalabilidad de un proyecto.

El uso de lo anteriormente mencionado permite simular entornos de infraestructura TI real, para poder comprender mejor las plataformas y adquirir habilidades de despliegue, configuración y pruebas de servicio en la nube.

Marco contextual

Este trabajo se realizó como proyecto final del Seminario AWS de la Universidad Remington ofrecido por la facultad de ingeniería, con la intención de integrar nuevos conocimientos y conceptos prácticos de diseño e implementación de redes en la nube.

Como contexto académico se resalta la búsqueda del crecimiento de las fortalezas hacia el manejo de nuevas tecnologías emergentes (Modernas), intentando adoptar una posición de mejora de los procesos continuos.

Estos ejercicios se realizaron en diferentes módulos al nivel GRATUITO de AWS, lo cual nos permitió diseñar nuestras propias instancias (LINUX Y WINDOWS) para publicar servicios en la web, administrarlos de manera versátil y explorar implementación de rendimiento como los Dockers; para cerrar todo este conocimiento técnico, nos dio la oportunidad de entender la importancia de la tecnología en la nube de los proyectos reales en el ámbito laboral y/o profesional.

Desarrollo e Implementación del aprendizaje

Diagrama de arquitectura

Representación gráfica de la red (EC2s, subredes, IPs públicas/privadas, grupos de seguridad, VPC, etc.)

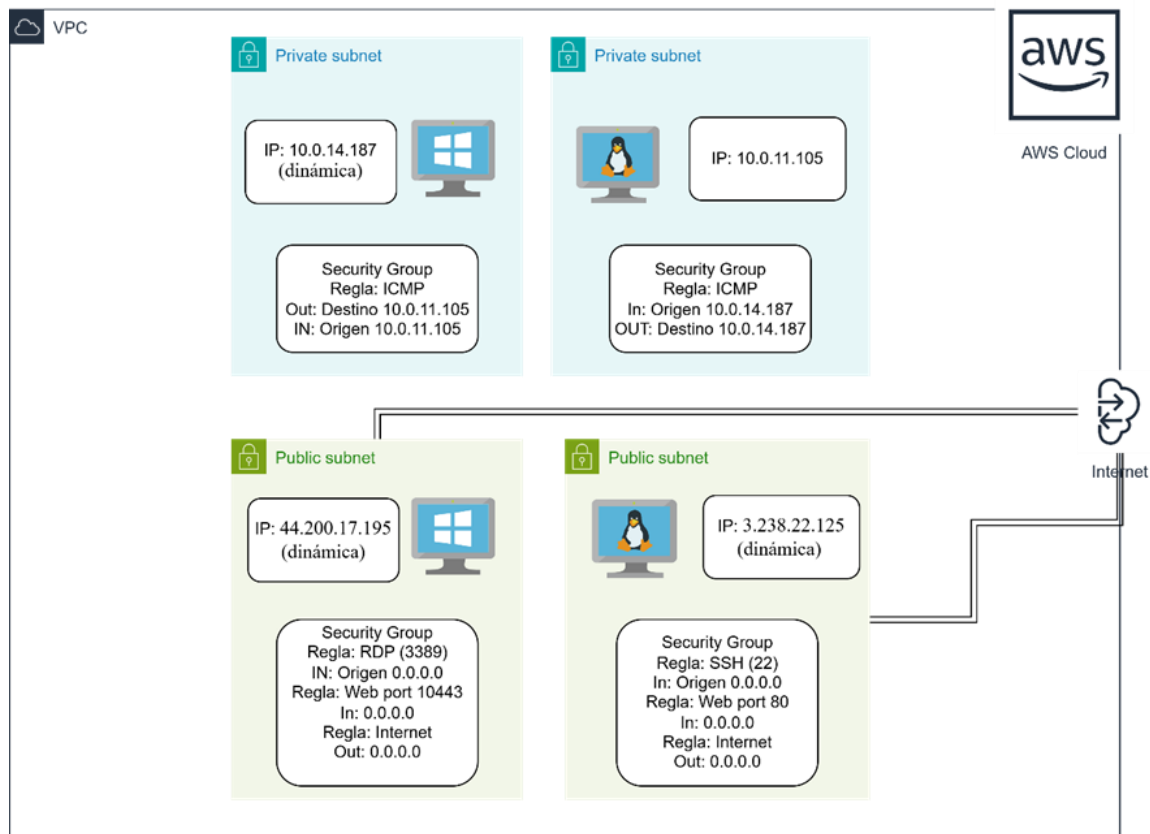


Ilustración 1

Descripción de la arquitectura

Se diseñó una arquitectura simple en la nube utilizando Amazon Web Services (AWS), compuesta por dos instancias EC2: una con sistema operativo Windows Server 2016 y otra con sistema Linux (Amazon Linux 2023). Ambas instancias se encuentran dentro de una misma red virtual (VPC), permitiendo la comunicación entre ellas. Esta red simula un entorno mixto, útil para pruebas de interoperabilidad, conectividad remota (RDP y SSH), y servicios web desde ambas plataformas.

Tipo de instancias usadas (**Linux: Amazon Linux. Windows: Windows Server**).

Se utilizaron dos tipos de instancias EC2 en AWS:

Instancia Linux: Amazon Linux 2023 (tipo t2.micro), utilizada como servidor web básico para alojar contenido HTML.

Instancia Windows: Windows Server 2016 (tipo t2.micro), utilizada para acceso remoto (RDP), utilizada como servidor web básico para alojar contenido HTML.

Ambas instancias están dentro del nivel gratuito de AWS, lo cual facilita su implementación sin incurrir en costos adicionales para pruebas básicas.

Justificación de las configuraciones de red

Las configuraciones de red fueron generadas automáticamente por AWS al momento de lanzar las instancias. Se creó una VPC por defecto con 2 subredes privadas y 2 públicas, lo que permitió asignar direcciones IP privadas y públicas a ambas instancias. Esto facilitó el acceso remoto mediante protocolos como SSH (puerto 22) para Linux y RDP (puerto 3389) para Windows desde Internet. El grupo de seguridad asociado fue configurado para permitir estas conexiones, garantizando el acceso necesario.

Configuraciones realizadas

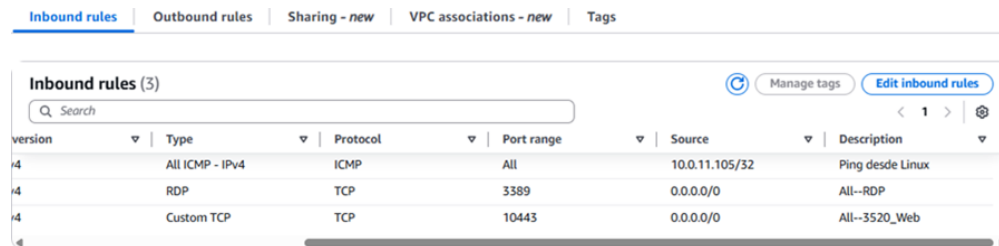
Pasos para crear las instancias EC2.

1. **Asignar nombre:** Se le da un nombre a la instancia para identificarla.
2. **Elegir SO y versión:** Se selecciona el sistema operativo (Linux) y su versión.
3. **Seleccionar tipo de instancia:** Se elige el tipo, como t2.micro para uso gratuito.
4. **Crear Key Pair:** Se genera una clave para acceder a la instancia de forma segura.
5. **Configurar red:**
 - **VPC:** Se crea Red virtual para la interconexión de las instancias.
 - **Grupo de seguridad:** Firewall que controla la interconexión a través de reglas.
 - **Reglas:** Se crean reglas para permitir los puertos 22 (SSH), 3389 (RDP) y 80 (HTTP).
6. **Asignar almacenamiento:** Se define el tamaño del disco, por defecto 8 GB.

Detalles de los Grupos de Seguridad (puertos abiertos: RDP, SSH, HTTP).

1. Windows

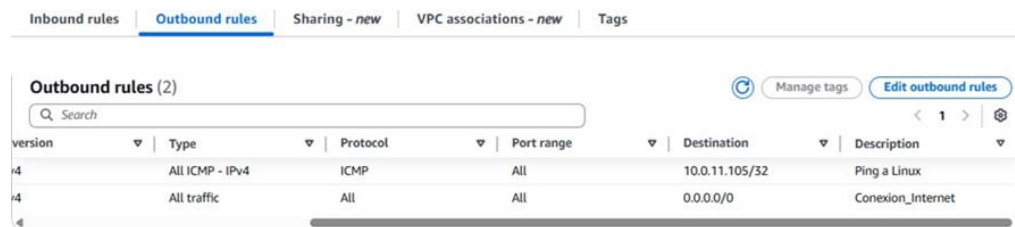
a. Inbound rules



version	Type	Protocol	Port range	Source	Description
4	All ICMP - IPv4	ICMP	All	10.0.11.105/32	Ping desde Linux
4	RDP	TCP	3389	0.0.0.0/0	All--RDP
4	Custom TCP	TCP	10443	0.0.0.0/0	All--3520_Web

Ilustración 2

b. Outbound rules

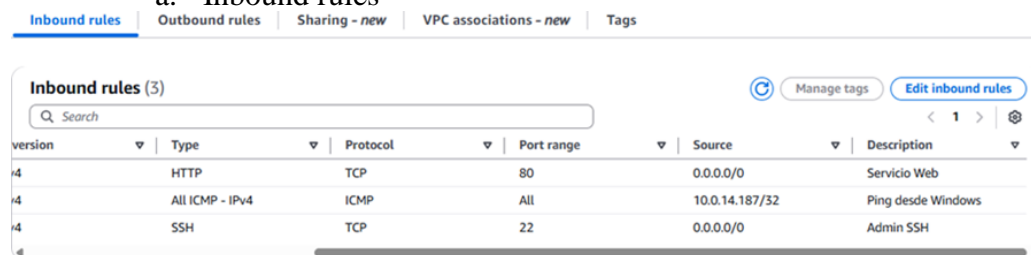


version	Type	Protocol	Port range	Destination	Description
4	All ICMP - IPv4	ICMP	All	10.0.11.105/32	Ping a Linux
4	All traffic	All	All	0.0.0.0/0	Conexion_Internet

Ilustración 3

2. Linux

a. Inbound rules



version	Type	Protocol	Port range	Source	Description
4	HTTP	TCP	80	0.0.0.0/0	Servicio Web
4	All ICMP - IPv4	ICMP	All	10.0.14.187/32	Ping desde Windows
4	SSH	TCP	22	0.0.0.0/0	Admin SSH

Ilustración 4

b. Outbound rules

Inbound rules | **Outbound rules** | Sharing - new | VPC associations - new | Tags

Outbound rules (2) Manage tags Edit outbound rules

Search

IP version	Type	Protocol	Port range	Destination	Description
IPv4	All ICMP - IPv4	ICMP	All	10.0.14.187/32	Ping hacia Windows
IPv4	All traffic	All	All	0.0.0.0/0	Conexion_Internet

Ilustración 5

Asignación de IPs públicas y privadas.

1. Windows

Privada y Pública

Instance summary for i-09b20d6e1760e21cc (Windows1) Connect Instance state Actions

Updated less than a minute ago

Instance ID i-09b20d6e1760e21cc	Public IPv4 address 44.200.3.67 open address	Private IPv4 addresses 10.0.14.187
IPv6 address -	Instance state Running	Public IPv4 DNS ec2-44-200-3-67.compute-1.amazonaws.com open address
Hostname type IP name: ip-10-0-14-187.ec2.internal	Private IP DNS name (IPv4 only) ip-10-0-14-187.ec2.internal	Elastic IP addresses -
Answer private resource DNS name -	Instance type t2.micro	AWS Compute Optimizer finding Opt-in to AWS Compute Optimizer for recommendations. Learn more
Auto-assigned IP address 44.200.3.67 [Public IP]	VPC ID vpc-0da55af12605e17fb (VPC-Seminario)	Auto Scaling Group name -
IAM Role -	Subnet ID subnet-047e3ea384217a80d (VPC-Seminario-subnet-public1-us-east-1a)	Managed false
IMDSv2 Required	Instance ARN arn:aws:ec2:us-east-1:081788263390:instance/i-09b20d6e1760e21cc	

Ilustración 6

2. Linux

Privada y Pública

Instance summary for i-03de5dd3edd6fa059 (Linux1) Connect Instance state Actions

Updated 1 minute ago

Instance ID i-03de5dd3edd6fa059	Public IPv4 address 13.219.228.11 open address	Private IPv4 addresses 10.0.11.105
IPv6 address -	Instance state Running	Public IPv4 DNS ec2-13-219-228-11.compute-1.amazonaws.com open address
Hostname type IP name: ip-10-0-11-105.ec2.internal	Private IP DNS name (IPv4 only) ip-10-0-11-105.ec2.internal	Elastic IP addresses -
Answer private resource DNS name -	Instance type t2.micro	AWS Compute Optimizer finding Opt-in to AWS Compute Optimizer for recommendations. Learn more
Auto-assigned IP address 13.219.228.11 [Public IP]	VPC ID vpc-0da55af12605e17fb (VPC-Seminario)	Auto Scaling Group name -
IAM Role -	Subnet ID subnet-047e3ea384217a80d (VPC-Seminario-subnet-public1-us-east-1a)	Managed false
IMDSv2 Required	Instance ARN arn:aws:ec2:us-east-1:081788263390:instance/i-03de5dd3edd6fa059	

Ilustración 7

Procedimiento de acceso

1. Windows

a) AWS

1. Ve a la consola de AWS en EC2 > Instances
2. Seleccionamos la instancia Windows
3. Obtener la contraseña de administrador
 - Clic en "Connect".
 - Clic en "RDP Client".
 - Clic en "Get Password".
 - Subimos el archivo .pem para descifrar la contraseña.
4. Copiar el usuario (Administrator) y la contraseña generada.

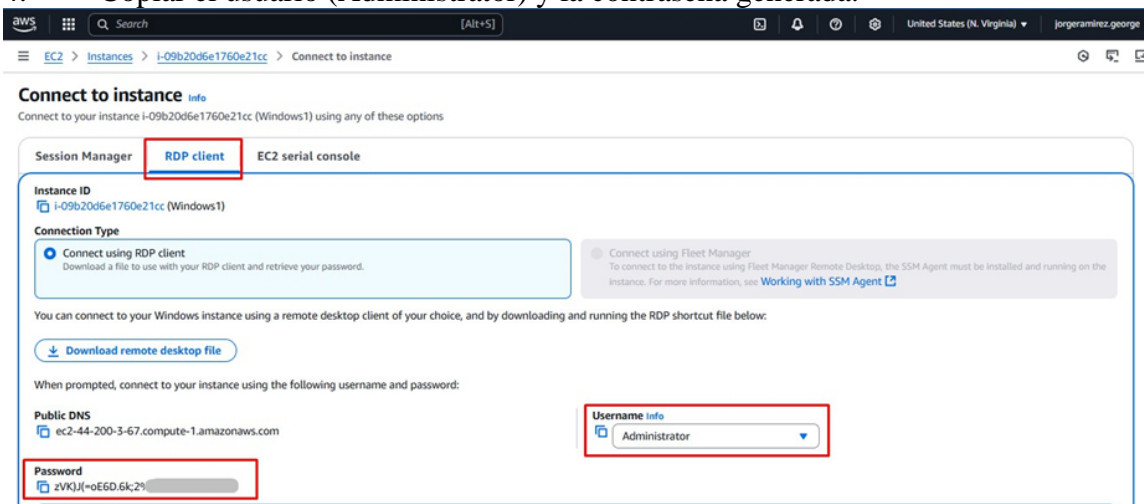
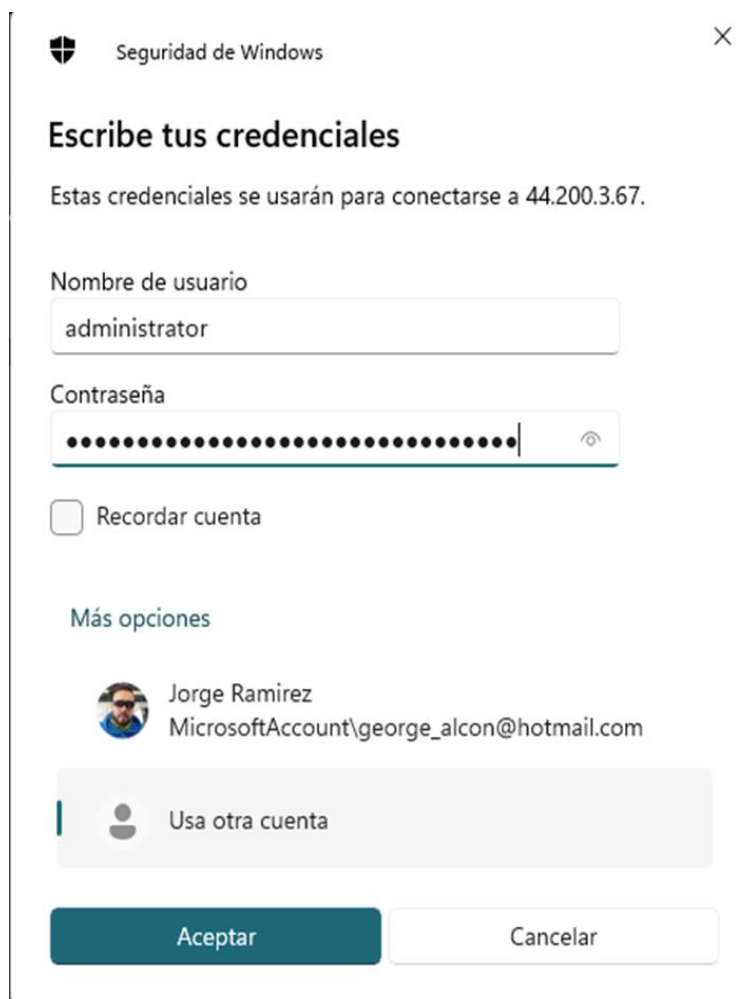


Ilustración 8

b) Cliente RDP

1. Abrir el programa Conexión a Escritorio Remoto de Windows
2. Escribimos la IP Pública de la instancia y clic en "Conectar".
3. Ingresamos el Usuario: *Administrator*
4. Ingresamos la contraseña: la que copiamos después de descifrarla en AWS.
5. Finalmente, clic en Aceptar.



The image shows a Windows Security dialog box titled "Seguridad de Windows" with a close button (X) in the top right corner. The main heading is "Escribe tus credenciales". Below this, it states "Estas credenciales se usarán para conectarse a 44.200.3.67." There are two input fields: "Nombre de usuario" containing "administrator" and "Contraseña" which is masked with dots and has a visibility icon on the right. Below the password field is a checkbox labeled "Recordar cuenta" which is unchecked. Under the heading "Más opciones", there is a user profile for "Jorge Ramirez" with a Microsoft Account email "MicrosoftAccount\george_alcon@hotmail.com". At the bottom, there is a button labeled "Usa otra cuenta" and two buttons at the very bottom: "Aceptar" (highlighted in dark teal) and "Cancelar".

Ilustración 9

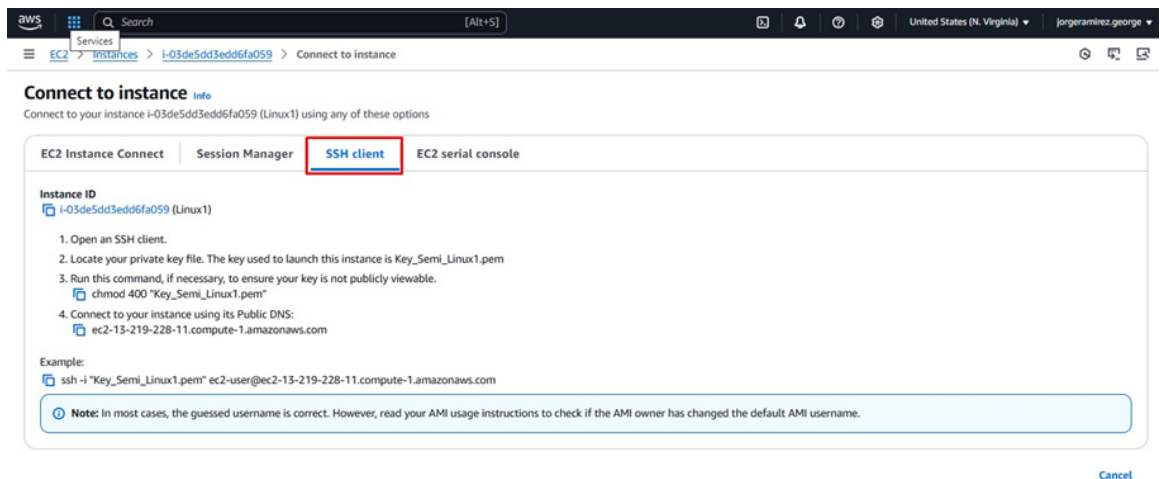


Ilustración 10

2. Linux

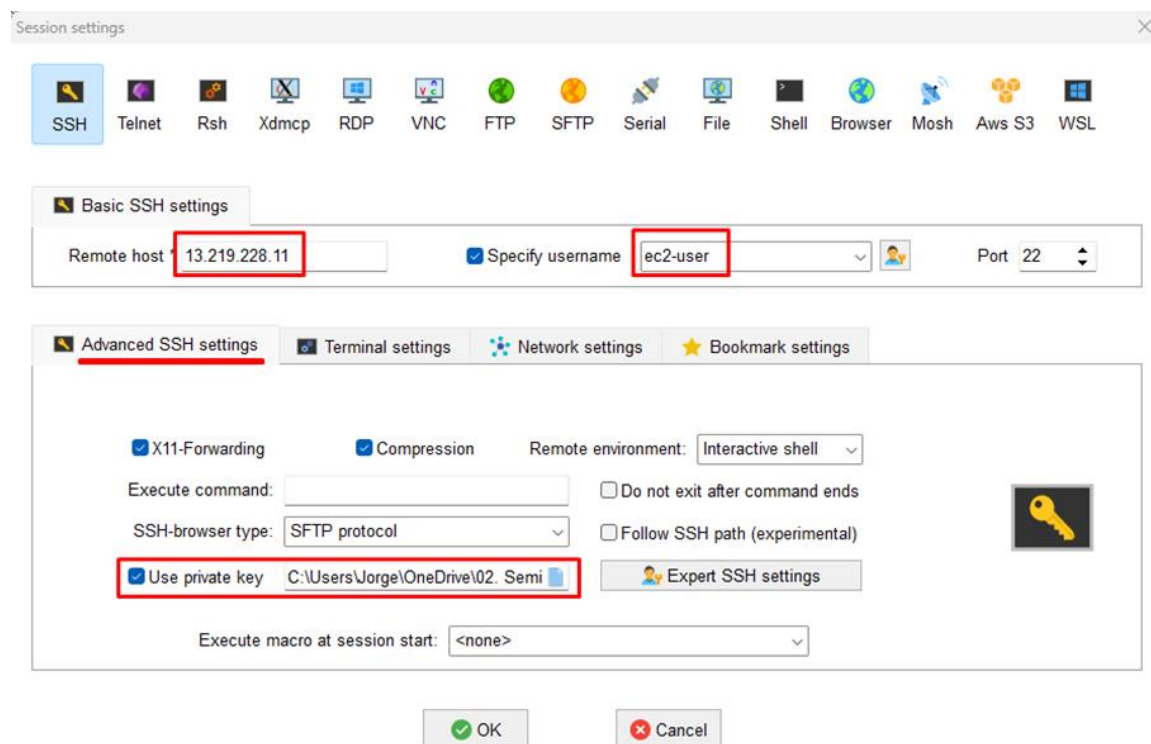
a) AWS

1. Ve a la consola de AWS en EC2 > Instances
2. Seleccionamos la instancia Linux
3. Clic en "Connect".
4. Clic en "SSH Client".
5. Copiar el usuario (ec2-user)



b) Cliente SSH

1. Abrir MobaXterm
 - Inicia el programa en el PC.
2. Crear una nueva sesión SSH
 - Clic en "Session" en la barra superior.
 - Seleccionamos la opción "SSH".
3. Configurar conexión SSH
 - En el campo Remote host, escribimos la IP pública de la instancia.
 - Marca "Specify username" y escribe el usuario correspondiente: ec2-user para Amazon Linux.
4. Cargar la clave privada (.pem)
 - En la parte izquierda, activar "Use private key".
 - Clic en el ícono de carpeta y selecciona el archivo.pem descargado desde AWS.
5. Conectar
 - Clic en "OK" para iniciar sesión en la instancia.

*Ilustración 11*

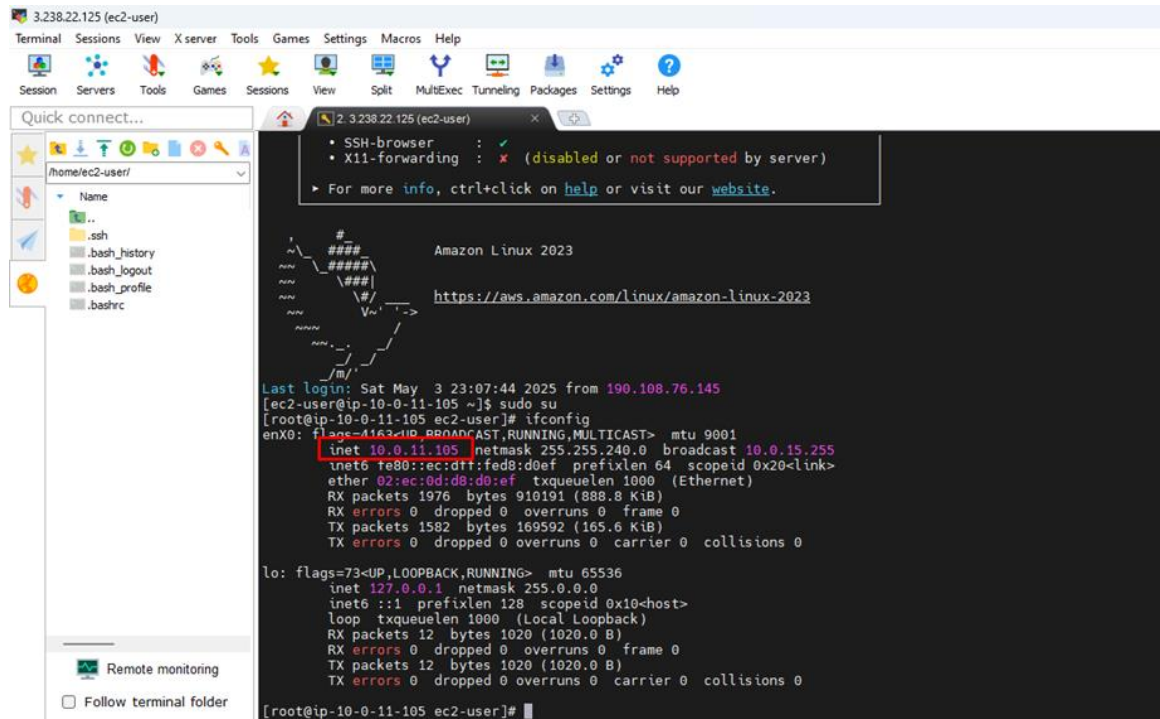


Ilustración 12

Configuración del servidor web

Pasos seguidos para instalar IIS en Windows Server.

1. Conectar por RDP:

a. Ingresa a la instancia EC2 de Windows desde AWS usando Remote Desktop (RDP).

2. Abrir el Administrador del Servidor:

a. Una vez dentro, haz clic en "Server Manager" (Administrador del servidor).

3. Agregar roles y características:

- a. En el panel de Server Manager, haz clic en:
- b. "Manage" → "Add Roles and Features".

4. Seleccionar instalación basada en roles

a. Marca la opción "Role-based or feature-based installation" y haz clic en Next.

5. Seleccionar servidor:

a. Selecciona tu servidor (ya aparecerá listado) y clic en Next.

6. Elegir rol: Web Server (IIS)

a. Marca la casilla "Web Server (IIS)".

b. Al salir la ventana emergente, haz clic en "Add Features" y luego en Next.

7. Confirmar e instalar

- Avanza con Next hasta llegar al resumen y haz clic en Install.
- Espera a que finalice la instalación (toma unos minutos).

8. Verificar IIS

- Abre un navegador en la instancia y entra a <http://localhost>

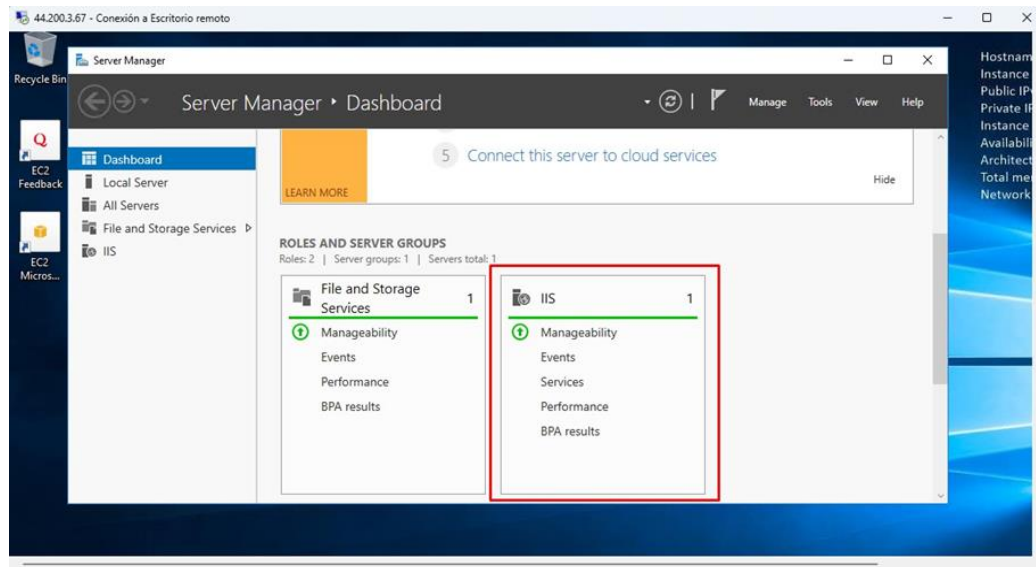


Ilustración 13

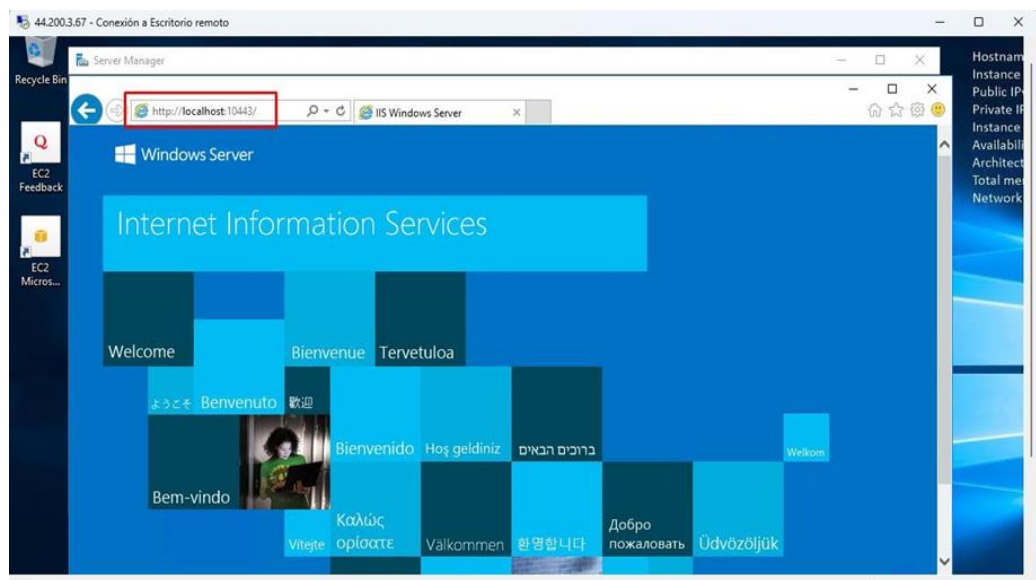


Ilustración 14



Ilustración 16

Pruebas básicas para verificar que los servidores web son accesibles desde Internet (captura de pantallas del navegador).

1. Servicio web publicado desde Windows: <http://44.200.3.67:10443/> (ejemplo)

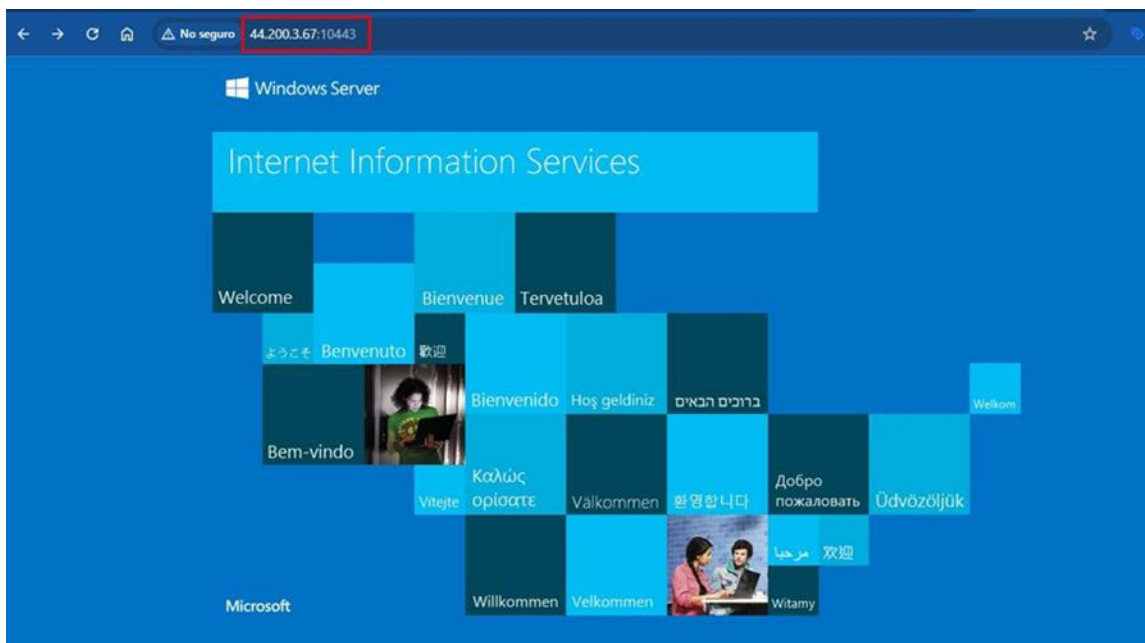


Ilustración 17

2. Servicio web publicado desde Linux: <http://13.219.228.11/> (ejemplo)



Ilustración 18

Pruebas de conectividad

Ping desde Windows a Linux

Para que el Linux responda Ping desde el Windows por la IP privada, es necesario crear una regla de seguridad saliente en el Windows y una entrante en el Linux.

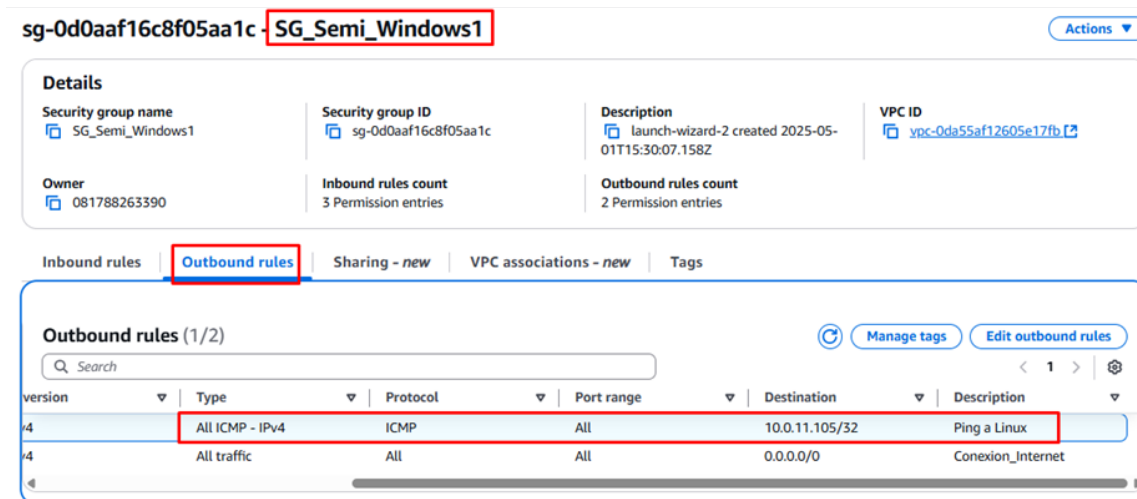


Ilustración 19

sg-01c88e5aa9715a305 - **SG_Semi_Linux1** Actions ▾

Details

Security group name SG_Semi_Linux1	Security group ID sg-01c88e5aa9715a305	Description launch-wizard-2 created 2025-05-01T19:04:06.681Z	VPC ID vpc-0da55af12605e17fb
Owner 081788263390	Inbound rules count 3 Permission entries	Outbound rules count 2 Permission entries	

Inbound rules | Outbound rules | Sharing - new | VPC associations - new | Tags

Inbound rules (1/3) Manage tags Edit inbound rules

Search

version	Type	Protocol	Port range	Source	Description
4	HTTP	TCP	80	0.0.0.0/0	Servicio Web
4	All ICMP - IPv4	ICMP	All	10.0.14.187/32	Ping desde Windows
4	SSH	TCP	22	0.0.0.0/0	Admin SSH

Ilustración 20

Ping desde Linux a Windows

Para que el Windows responda Ping desde el Linux por la IP privada, es necesario crear una regla de seguridad saliente en el Linux y una entrante en el Windows.

sg-01c88e5aa9715a305 - **SG_Semi_Linux1** Actions ▾

Details

Security group name SG_Semi_Linux1	Security group ID sg-01c88e5aa9715a305	Description launch-wizard-2 created 2025-05-01T19:04:06.681Z	VPC ID vpc-0da55af12605e17fb
Owner 081788263390	Inbound rules count 3 Permission entries	Outbound rules count 2 Permission entries	

Inbound rules | **Outbound rules** | Sharing - new | VPC associations - new | Tags

Outbound rules (2) Manage tags Edit outbound rules

Search

version	Type	Protocol	Port range	Destination	Description
4	All ICMP - IPv4	ICMP	All	10.0.14.187/32	Ping hacia Windows
4	All traffic	All	All	0.0.0.0/0	Conexion_Internet

sg-0d0aaf16c8f05aa1c - **SG_Semi_Windows1** Actions ▾

Details

Security group name SG_Semi_Windows1	Security group ID sg-0d0aaf16c8f05aa1c	Description launch-wizard-2 created 2025-05-01T15:30:07.158Z	VPC ID vpc-0da55af12605e17fb
Owner 081788263390	Inbound rules count 3 Permission entries	Outbound rules count 2 Permission entries	

Inbound rules | **Outbound rules** | Sharing - new | VPC associations - new | Tags

Inbound rules (1/3) Manage tags Edit inbound rules

Search

version	Type	Protocol	Port range	Source	Description
4	All ICMP - IPv4	ICMP	All	10.0.11.105/32	Ping desde Linux
4	RDP	TCP	3389	0.0.0.0/0	All-RDP
4	Custom TCP	TCP	10443	0.0.0.0/0	All-3520_Web

Ilustración 21

Adicionalmente, para que el Windows responda a Ping, se requiere crear una regla en el Firewall interno de Windows.

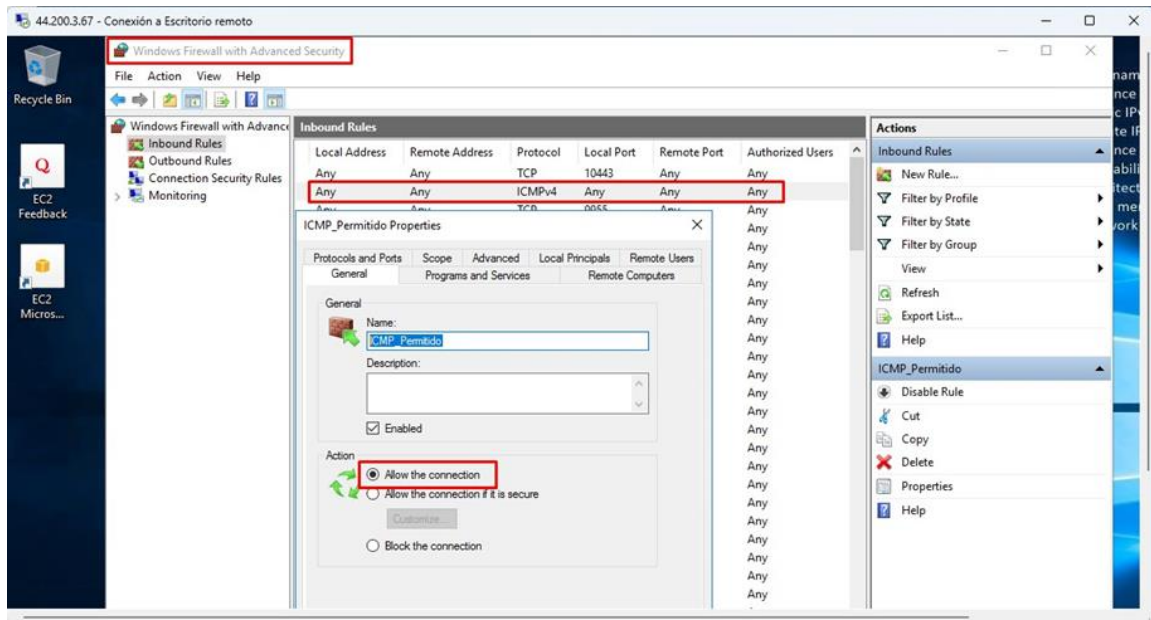


Ilustración 22

PRUEBAS DOCKER

Que es Docker:

Este es un servicio o librería que permite crear, ejecutar y administrar servicios (aplicaciones) dentro de contenedores, estos terminan siendo entornos ligeros y portables que se apropian de lo necesario para ejecutar un servicio: sea código, algunas dependencias, librerías y/o configuraciones.

NOTA: Generalmente los contenedores se componen de AMIS o IMG

Qué son las Máquinas Virtuales (VMs):

Estas son las que se ejecutan con herramientas externas como VirtualBox, VMware o directamente en Linux con comandos Sudo, funcionan emulando un sistema operativo, lo que a la final genera más consumo de recursos.

Diferencias clave entre Docker y Máquinas virtuales

Características para diferenciar ambas tecnologías

Características	Docker (Contenedores)	Máquinas Virtuales (VMs)
Nivel de virtualización	A nivel del sistema operativo (usa el mismo kernel)	A nivel del hardware
Tamaño	Ligero (MB)	Pesado (GB)
Arranque	Rápido (segundos)	Lento (minutos)
Aislamiento	Moderado (comparten kernel)	Alto (cada VM es completamente aislada)
Consumo de recursos	Bajo (sin duplicar SO)	Alto (cada VM necesita RAM y CPU para el SO)
Portabilidad	Muy portable (misma imagen funciona en varios SO)	Menos portable (depende del hipervisor/SO)

Ventajas del Docker frente a las Máquinas virtuales.

1. **El Rendimiento más alto:**
Al no tener que arrancar un sistema operativo completo, los contenedores consumen menos CPU, memoria y disco.
2. **El Despliegue más rápido:**
Los contenedores pueden iniciarse en segundos.
3. **La Mayor densidad:**
Deja correr más contenedores en un mismo host, mientras que las VM's son más limitadas por el consumo de recursos tan elevado.
4. **La Portabilidad:**
Los contenedores funcionan igual en desarrollo, pruebas y producción (siempre que Docker esté instalado).
5. **La Menor sobrecarga administrativa:**
Menos mantenimiento de sistemas operativos internos.

Procedimiento de un Docker

- 1) Instalamos Docker con el comando “`dnf install docker`”.

```
[ec2-user@ip-10-0-24-241 ~]$ sudo su
[root@ip-10-0-24-241 ec2-user]# dnf install docker
```

Ilustración 23

- 2) Confirmamos el estado del servicio Docker con el comando “`systemctl status docker`” y buscamos que esté (activo), en el siguiente estado no lo visualizamos de esa forma por lo tanto procedemos con el punto #3.

```
[root@ip-10-0-24-241 ec2-user]# 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 24

- 3) A continuación, arrancamos el servicio con el comando “`systemctl start docker`”.

```
[root@ip-10-0-24-241 ec2-user]# systemctl start docker
```

Ilustración 25

- 4) Volvemos a validar el estado del servicio con el mismo comando del punto #2 “`systemctl status docker`”.

```
Loaded: loaded (/usr/lib/systemd/system/docker.service; disabled; preset: disabled)
Active: active (running) since Thu 2025-05-15 21:20:31 UTC; 45s ago
Process: 1 docker socket
```

Ilustración 26

- 5) Para crear la información interna del Docker debemos generarla a través de dockerfile (las IMG), ingresamos dentro de las IMG y extraemos su “nombre”.
- 6) Apuntamos el `docker pull httpd` con el comando al frente de los datos de la IMG.



Ilustración 27

- 7) Para descargarlo escribimos el comando obtenido en la consola LINUX.

```
[root@ip-10-0-24-241 ec2-user]# 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
```

Ilustración 28

- 8) Validamos que si se haya descargado con *docker images*.

```
[root@ip-10-0-24-241 ec2-user]# docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
httpd latest 0208f149a449 3 months ago 148MB
```

Ilustración 29

- 9) Descargamos otra imagen también de Nginx.

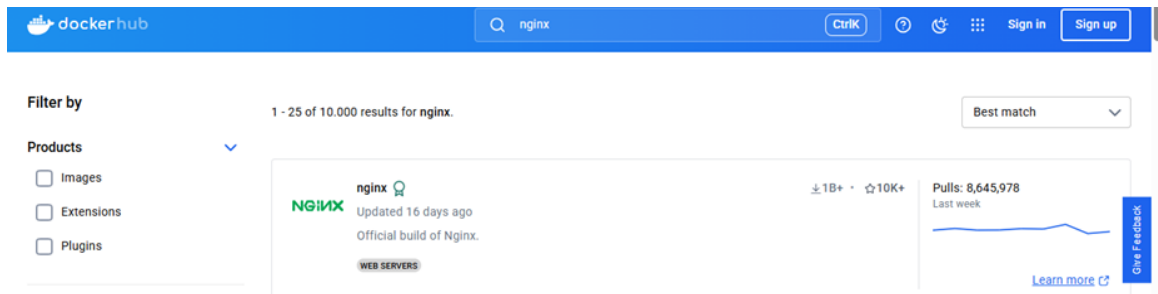


Ilustración 30

- 10) Miramos el comando de la misma manera que el anterior y lo ejecutamos.



Ilustración 31

```
[root@ip-10-0-24-241 ec2-user]# docker pull nginx
Using default tag: latest
latest: Pulling from library/nginx
254e724d7786: Already exists
913115292750: Pull complete
3e544d53ce49: Pull complete
4f21ed9ac0c0: Pull complete
d38f2ef2d6f2: Pull complete
40a6e9f4e456: Pull complete
d3dc5ec71e9d: Pull complete
Digest: sha256:c15da6c91de8d2f436196f3a768483ad32c258ed4e1beb3d367a27ed67253e66
Status: Downloaded newer image for nginx:latest
docker.io/library/nginx:latest
```

Ilustración 32

- 11) Ya con las imágenes descargadas ejecutamos “*docker ps*” para ver los contenedores creados.

```
[root@ip-10-0-24-241 ec2-user]# docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES
```

Ilustración 33

- 12) Si no hay contenedores lo creamos como un “*demonio*” que es una ejecución que está en segundo plano, de la siguiente forma:
 -dit (esta es una variable) --name -p (definimos el puerto)
 estas nos retornan un hash de verificación.

```
[root@ip-10-0-24-241 ec2-user]# docker run -dit --name contenedorEjemplo -p 8083:80 httpd
24e23e193bc1599b7bc673ec254fd41606e8c0e8b2dc7a98763bbe2dd56c78fa
[root@ip-10-0-24-241 ec2-user]# docker run -dit --name contenedorTrabajo -p 8084:80 httpd
fad1205d7d187ebef9f30a2321c37adc3f9168d09c35f529e748cd53535c81b9
[root@ip-10-0-24-241 ec2-user]# docker run -dit --name contenedorEntrega -p 8085:80 httpd
711c87935496fa9d7f6841a4705ea47a984363d6d010d7916aa89d11eaaa70b6
```

Ilustración 34

- 13) Visualizamos el contenedor con *docker ps*.

```
[root@ip-10-0-24-241 ec2-user]# docker run -dit --name contenedorEjemplo -p 8083:80 httpd
24e23e193bc1599b7bc673ec254fd41606e8c0e8b2dc7a98763bbe2dd56c78fa
[root@ip-10-0-24-241 ec2-user]# docker run -dit --name contenedorTrabajo -p 8084:80 httpd
fad1205d7d187ebef9f30a2321c37adc3f9168d09c35f529e748cd53535c81b9
[root@ip-10-0-24-241 ec2-user]# docker run -dit --name contenedorEntrega -p 8085:80 httpd
711c87935496fa9d7f6841a4705ea47a984363d6d010d7916aa89d11eaaa70b6
```

Ilustración 35

14) Configuramos el puerto 8080 para poder tener accesos desde cualquier IP.

EC2 > Security Groups > sg-05b23b49a56b2b0f7 - SG-ServerLinux > Edit inbound rules

Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

Security group rule ID	Type <small>Info</small>	Protocol <small>Info</small>	Port range <small>Info</small>	Source <small>Info</small>	Description - optional <small>Info</small>	Delete
sgr-0d1a1e97998fd3ea1	Custom TCP	TCP	8001	Cus... <input type="text" value="0.0.0.0/0"/>		Delete
sgr-01dcaa452630555ed	Custom TCP	TCP	8026	Cus... <input type="text" value="0.0.0.0/0"/>		Delete
sgr-0b0e0d051880a1ef83	Custom TCP	TCP	8002	Cus... <input type="text" value="0.0.0.0/0"/>		Delete
sgr-068824f9c7271d203	Custom TCP	TCP	8084	Cus... <input type="text" value="0.0.0.0/0"/>		Delete
sgr-0ef89f9c46ad327ba	Custom TCP	TCP	8086	Cus... <input type="text" value="0.0.0.0/0"/>		Delete

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

▼ Security details

IAM Role -	Owner ID 075003031404	Launch time Thu May 15 2025 13:01:33 GMT-0600 (hora estándar central)
Security groups sg-05b23b49a56b2b0f7 (SG-ServerLinux)		

► **Inbound rules**

[Inbound rules](#) | [Outbound rules](#) | [Sharing - new](#) | [VPC associations - new](#) | [Tags](#)

Inbound rules (2)

[Manage tags](#) [Edit inbound rules](#)

<input type="checkbox"/>	Name	Security group rule ID	IP version	Type	Protocol
<input type="checkbox"/>	-	sgr-0bacbca359ed2892f	IPv4	SSH	TCP
<input type="checkbox"/>	-	sgr-04e214e4711bdff65	IPv4	Custom TCP	TCP

Ilustración 36

15) Validamos en Internet con cualquiera de los puertos creados.

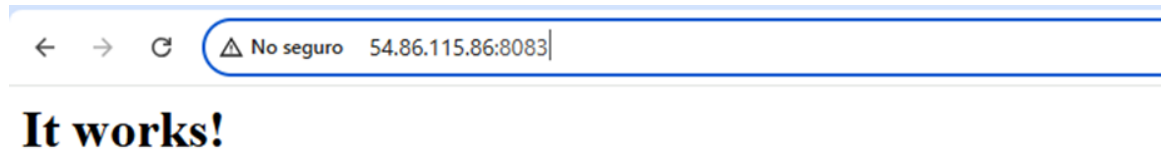


Ilustración 37

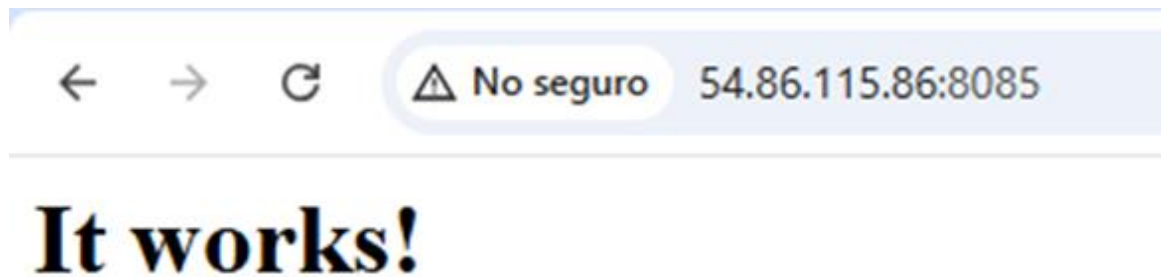


Ilustración 38

16) Confirmamos el consumo de uso de CPU y RAM con el comando “top”.

```
[root@ip-10-0-24-241 ec2-user]# top
top - 22:26:40 up 3:25, 2 users, load average: 0.00, 0.00, 0.00
Tasks: 140 total, 2 running, 138 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.6 us, 0.6 sy, 0.0 ni, 93.1 id, 0.0 wa, 0.0 hi, 0.0 si, 5.6 st
MiB Mem : 949.4 total, 84.7 free, 315.5 used, 549.2 buff/cache
MiB Swap: 0.0 total, 0.0 free, 0.0 used, 466.3 avail Mem
```

Ilustración 39

```
[root@ip-10-0-24-241 ec2-user]# top
top - 04:30:52 up 9:29, 2 users, load average: 0.03, 0.03, 0.00
Tasks: 355 total, 1 running, 354 sleeping, 0 stopped, 0 zombie
%Cpu(s): 3.5 us, 1.3 sy, 0.0 ni, 92.9 id, 0.0 wa, 0.0 hi, 0.0 si, 2.3 st
MiB Mem : 949.4 total, 63.0 free, 743.9 used, 142.5 buff/cache
MiB Swap: 0.0 total, 0.0 free, 0.0 used, 57.2 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
33809	root	20	0	1877116	7656	2216	S	1.3	0.8	0:00.88	containerd-shim
35535	root	20	0	1877116	8000	2468	S	0.7	0.8	0:01.14	containerd-shim
35824	root	20	0	1877372	8004	2560	S	0.7	0.8	0:00.68	containerd-shim
11742	root	20	0	2633324	49876	7536	S	0.3	5.1	0:11.04	dockerd
34131	root	20	0	5864	1164	316	S	0.3	0.1	0:00.14	httpd
35246	root	20	0	1877372	8012	2896	S	0.3	0.8	0:00.79	containerd-shim
36112	root	20	0	1877372	7684	2252	S	0.3	0.8	0:00.78	containerd-shim
36406	root	20	0	1877372	8008	2568	S	0.3	0.8	0:01.06	containerd-shim
36713	root	20	0	1887616	7792	2320	S	0.3	0.8	0:01.15	containerd-shim
44585	root	20	0	224300	3560	2588	R	0.3	0.4	0:00.39	top
1	root	20	0	108476	13060	4900	S	0.0	1.3	0:04.13	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kthreadd
3	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	rcu_gp
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	rcu_par_gp
5	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	slub_flushwq
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	netns
8	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/0:0H-events_highpri
10	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	mm_percpu_wq
11	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_kthread
12	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_rude_kthread
13	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_trace_kthread
14	root	20	0	0	0	0	S	0.0	0.0	0:00.91	ksoftirqd/0
15	root	20	0	0	0	0	I	0.0	0.0	0:00.86	rcu_preempt
16	root	rt	0	0	0	0	S	0.0	0.0	0:00.15	migration/0
18	root	20	0	0	0	0	S	0.0	0.0	0:00.00	cpuhp/0
20	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kdevtmpfs
21	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	inet_frag_wq
22	root	20	0	0	0	0	S	0.0	0.0	0:00.02	kauditd
23	root	20	0	0	0	0	S	0.0	0.0	0:00.02	khungtaskd

Ilustración 40

```
[root@ip-10-0-24-241 ec2-user]# top
top - 04:32:04 up 9:30, 2 users, load average: 0.01, 0.02, 0.00
Tasks: 355 total, 1 running, 354 sleeping, 0 stopped, 0 zombie
%Cpu(s): 1.3 us, 2.0 sy, 0.0 ni, 95.7 id, 0.0 wa, 0.0 hi, 0.3 si, 0.7 st
MiB Mem : 949.4 total, 63.0 free, 743.3 used, 143.1 buff/cache
MiB Swap: 0.0 total, 0.0 free, 0.0 used, 57.5 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
11731	root	20	0	1745972	29876	6076	S	0.3	3.1	0:26.10	containerd
37605	root	20	0	5864	1168	316	S	0.3	0.1	0:00.11	httpd
42195	root	20	0	1877372	8124	2776	S	0.3	0.8	0:00.69	containerd-shim
44585	root	20	0	224300	3560	2588	R	0.3	0.4	0:00.58	top
1	root	20	0	108476	13060	4900	S	0.0	1.3	0:04.13	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kthreadd
3	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	rcu_gp
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	rcu_par_gp
5	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	slub_flushwq
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	netns
8	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/0:0H-events_highpri
10	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	mm_percpu_wq
11	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_kthread
12	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_rude_kthread
13	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_trace_kthread
14	root	20	0	0	0	0	S	0.0	0.0	0:00.91	ksoftirqd/0
15	root	20	0	0	0	0	I	0.0	0.0	0:00.86	rcu_preempt
16	root	rt	0	0	0	0	S	0.0	0.0	0:00.15	migration/0
18	root	20	0	0	0	0	S	0.0	0.0	0:00.00	cpuhp/0
20	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kdevtmpfs
21	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	inet_frag_wq
22	root	20	0	0	0	0	S	0.0	0.0	0:00.02	kauditd
23	root	20	0	0	0	0	S	0.0	0.0	0:00.02	khungtaskd
24	root	20	0	0	0	0	S	0.0	0.0	0:00.00	oom_reaper
27	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	wriTeback
28	root	20	0	0	0	0	S	0.0	0.0	0:00.78	kcompactd0
29	root	39	19	0	0	0	S	0.0	0.0	0:00.00	khugepaged
30	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	cryptd
31	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kintegrityd

Ilustración 41

```
[root@ip-10-0-24-241 ec2-user]# top
top - 04:32:37 up 9:30, 2 users, load average: 0.00, 0.02, 0.00
Tasks: 355 total, 1 running, 354 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.3 us, 0.0 sy, 0.0 ni, 94.0 id, 0.0 wa, 0.0 hi, 0.0 st, 5.7 st
MiB Mem : 949.4 total, 62.9 free, 743.3 used, 143.1 buff/cache
MiB Swap: 0.0 total, 0.0 free, 0.0 used, 57.5 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
44585	root	20	0	224300	3560	2588	R	0.3	0.4	0:00.66	top
1	root	20	0	108476	13060	4900	S	0.0	1.3	0:04.13	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kthreadd
3	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	rcu_gp
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	rcu_par_gp
5	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	slub_flushwq
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	netns
8	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/0:0H-events_highpri
10	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	mm_percpu_wq
11	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_kthread
12	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_rude_kthread
13	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_trace_kthread
14	root	20	0	0	0	0	S	0.0	0.0	0:00.91	ksoftirqd/0
15	root	20	0	0	0	0	I	0.0	0.0	0:00.86	rcu_preempt
16	root	rt	0	0	0	0	S	0.0	0.0	0:00.15	migration/0
18	root	20	0	0	0	0	S	0.0	0.0	0:00.00	cpuhp/0
20	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kdevtmpfs
21	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	inet_frag_wq
22	root	20	0	0	0	0	S	0.0	0.0	0:00.02	kauditd
23	root	20	0	0	0	0	S	0.0	0.0	0:00.02	khungtaskd
24	root	20	0	0	0	0	S	0.0	0.0	0:00.00	oom_reaper
27	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	writeback
28	root	20	0	0	0	0	S	0.0	0.0	0:00.78	kcompactd0
29	root	39	19	0	0	0	S	0.0	0.0	0:00.00	khugepaged
30	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	cryptd
31	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kintegrityd
32	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kblockd
33	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	blkcg_punt_bio
34	root	20	0	0	0	0	S	0.0	0.0	0:00.00	xen-balloon

Ilustración 42

```
[root@ip-10-0-24-241 ec2-user]# top
top - 04:33:07 up 9:31, 2 users, load average: 0.00, 0.02, 0.00
Tasks: 355 total, 1 running, 354 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.0 us, 0.0 sy, 0.0 ni, 98.4 id, 0.0 wa, 0.0 hi, 0.3 st, 1.3 st
MiB Mem : 949.4 total, 62.9 free, 743.3 used, 143.1 buff/cache
MiB Swap: 0.0 total, 0.0 free, 0.0 used, 57.5 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
44585	root	20	0	224300	3560	2588	R	0.3	0.4	0:00.74	top
1	root	20	0	108476	13060	4900	S	0.0	1.3	0:04.14	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kthreadd
3	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	rcu_gp
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	rcu_par_gp
5	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	slub_flushwq
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	netns
8	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/0:0H-events_highpri
10	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	mm_percpu_wq
11	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_kthread
12	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_rude_kthread
13	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_tasks_trace_kthread
14	root	20	0	0	0	0	S	0.0	0.0	0:00.91	ksoftirqd/0
15	root	20	0	0	0	0	I	0.0	0.0	0:00.86	rcu_preempt
16	root	rt	0	0	0	0	S	0.0	0.0	0:00.15	migration/0
18	root	20	0	0	0	0	S	0.0	0.0	0:00.00	cpuhp/0
20	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kdevtmpfs
21	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	inet_frag_wq
22	root	20	0	0	0	0	S	0.0	0.0	0:00.02	kauditd
23	root	20	0	0	0	0	S	0.0	0.0	0:00.02	khungtaskd
24	root	20	0	0	0	0	S	0.0	0.0	0:00.00	oom_reaper
27	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	writeback
28	root	20	0	0	0	0	S	0.0	0.0	0:00.78	kcompactd0
29	root	39	19	0	0	0	S	0.0	0.0	0:00.00	khugepaged
30	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	cryptd
31	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kintegrityd
32	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kblockd
33	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	blkcg_punt_bio
34	root	20	0	0	0	0	S	0.0	0.0	0:00.00	xen-balloon

Ilustración 43

17) Vemos el uso que tienen los contenedores con el comando “*docker stats*”.

```
[root@ip-10-0-24-241 ec2-user]# docker stats
```

Ilustración 44

```
[root@ip-10-0-24-241 ec2-user]# docker stats
CONTAINER ID   NAME                CPU %     MEM USAGE / LIMIT   MEM %     NET I/O       BLOCK I/O   PIDS
4e0a81e4ec1d   contenedorN30      0.00%    6.488MiB / 949.4MiB 0.68%     1.57kB / 0B   6.27MB / 4.1kB 82
102fc46e02eb   contenedorN29      0.00%    6.176MiB / 949.4MiB 0.65%     1.57kB / 0B   5.98MB / 4.1kB 82
8727ed404426   contenedorN28      0.00%    6.109MiB / 949.4MiB 0.64%     1.57kB / 0B   4.41MB / 4.1kB 82
8f82690777f3   contenedorN27      0.00%    7.695MiB / 949.4MiB 0.81%     3.48kB / 1.78kB 2.15MB / 4.1kB 109
15f397a9ae9    contenedorN26      0.00%    6.008MiB / 949.4MiB 0.63%     1.57kB / 0B   287kB / 4.1kB 82
6f5af1d9c303   contenedorN25      0.00%    6.066MiB / 949.4MiB 0.64%     1.57kB / 0B   721kB / 4.1kB 82
652ce43c9cb2   contenedorN24      0.00%    6.043MiB / 949.4MiB 0.64%     1.57kB / 0B   102kB / 4.1kB 82
24dc5a9b073b   contenedorN23      0.00%    6.055MiB / 949.4MiB 0.64%     1.57kB / 0B   45.1kB / 4.1kB 82
0509723d6f3f   contenedorN22      0.00%    6.008MiB / 949.4MiB 0.63%     1.67kB / 432B  4.1kB / 4.1kB 82
84749e14fced   contenedorN21      0.00%    6.023MiB / 949.4MiB 0.63%     1.57kB / 0B   262kB / 4.1kB 82
698c38d7f3e3   contenedorN20      0.00%    6.031MiB / 949.4MiB 0.64%     1.57kB / 0B   57.3kB / 4.1kB 82
f6838a4e2cf3   contenedorN19      0.00%    6.023MiB / 949.4MiB 0.63%     1.57kB / 0B   69.6kB / 4.1kB 82
eef8e72a0f92   contenedorN18      0.00%    6.035MiB / 949.4MiB 0.64%     1.57kB / 0B   20.5kB / 4.1kB 82
13b270adc526   contenedorN17      0.00%    6.039MiB / 949.4MiB 0.64%     1.57kB / 0B   12.3kB / 4.1kB 82
bb028eadf533   contenedorN16      0.00%    6.004MiB / 949.4MiB 0.63%     1.57kB / 0B   28.7kB / 4.1kB 82
ef0e4d350815   contenedorN15      0.00%    6.016MiB / 949.4MiB 0.63%     1.57kB / 0B   0B / 4.1kB 82
d1a99e7d2ede   contenedorN14      0.00%    6.047MiB / 949.4MiB 0.64%     1.57kB / 0B   57.3kB / 4.1kB 82
27c34cbb763d   contenedorN13      0.00%    6.051MiB / 949.4MiB 0.64%     1.57kB / 0B   0B / 4.1kB 82
81f53f4142af   contenedorN12      0.00%    6.051MiB / 949.4MiB 0.64%     1.57kB / 0B   8.19kB / 4.1kB 82
7fe276f2271c   contenedorN11      0.00%    6.008MiB / 949.4MiB 0.63%     1.57kB / 0B   0B / 4.1kB 82
b6f2f86dab3c   contenedorN6       0.00%    8.457MiB / 949.4MiB 0.89%     3.52kB / 1.82kB 2.99MB / 4.1kB 109
1f19bdad9188   contenedorN4       0.00%    6.008MiB / 949.4MiB 0.63%     1.71kB / 474B  53.2kB / 4.1kB 82
1871adb949bd   contenedorN3       0.00%    6.012MiB / 949.4MiB 0.63%     1.57kB / 0B   115kB / 4.1kB 82
7fbf0876889e   contenedorN2       0.00%    6.035MiB / 949.4MiB 0.64%     1.61kB / 0B   12.3kB / 4.1kB 82
1a2102c2fe38   contenedorN1       0.00%    6.109MiB / 949.4MiB 0.64%     2.33kB / 642B  0B / 4.1kB 82
4c4fd2dc6bcd   contenedorN10      0.00%    6.035MiB / 949.4MiB 0.64%     2.35kB / 1.84kB 4.1kB / 4.1kB 82
43ba6c239d02   contenedorN9       0.00%    6.008MiB / 949.4MiB 0.63%     1.61kB / 0B   0B / 4.1kB 82
4cbbfa8a5e55   contenedorN8       0.00%    6.02MiB / 949.4MiB  0.63%     1.75kB / 474B  0B / 4.1kB 82
2261a43356e2   contenedorN7       0.00%    6.031MiB / 949.4MiB 0.64%     1.68kB / 0B   0B / 4.1kB 82
9940cc055f4b   contenedorN5       0.00%    6.023MiB / 949.4MiB 0.63%     1.76kB / 142B  0B / 4.1kB 82
711c87935496   contenedorEntrega  0.00%    7.648MiB / 949.4MiB 0.81%     5.26kB / 2.12kB 299kB / 4.1kB 109
fad1205d7d18   contenedorTrabajo  0.00%    7.742MiB / 949.4MiB 0.82%     6.95kB / 3.59kB 614kB / 4.1kB 109
24e23e193bc1   contenedorEjemplo  0.00%    7.723MiB / 949.4MiB 0.81%     7.44kB / 5.35kB 172kB / 4.1kB 109
1014b072bd34   app1               0.00%    7.844MiB / 949.4MiB 0.83%     15.8kB / 16kB  3.21MB / 4.1kB 109
CONTAINER ID   NAME                CPU %     MEM USAGE / LIMIT   MEM %     NET I/O       BLOCK I/O   PIDS
```

Ilustración 45

Pasos Para Detener el Docker

- 1) Buscamos el Id del contenedor con el comando “*docker ps*”.

```
[root@ip-10-0-24-241 ec2-user]# docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
4e0a81e4ec1d	httpd	"httpd-foreground"	55 minutes ago	Up 55 minutes	0.0.0.0:8030->80/tcp, :::8030->80/tcp	contenedorN30
102fc46e02ab	httpd	"httpd-foreground"	55 minutes ago	Up 55 minutes	0.0.0.0:8029->80/tcp, :::8029->80/tcp	contenedorN29
8727ed04426	httpd	"httpd-foreground"	55 minutes ago	Up 55 minutes	0.0.0.0:8028->80/tcp, :::8028->80/tcp	contenedorN28
8f82690777f3	httpd	"httpd-foreground"	55 minutes ago	Up 55 minutes	0.0.0.0:8027->80/tcp, :::8027->80/tcp	contenedorN27
15f397a9aea9	httpd	"httpd-foreground"	55 minutes ago	Up 55 minutes	0.0.0.0:8026->80/tcp, :::8026->80/tcp	contenedorN26
6f5af1d9c303	httpd	"httpd-foreground"	55 minutes ago	Up 55 minutes	0.0.0.0:8025->80/tcp, :::8025->80/tcp	contenedorN25
652ce43c9cb2	httpd	"httpd-foreground"	55 minutes ago	Up 55 minutes	0.0.0.0:8024->80/tcp, :::8024->80/tcp	contenedorN24
24dc5a9b073b	httpd	"httpd-foreground"	56 minutes ago	Up 55 minutes	0.0.0.0:8023->80/tcp, :::8023->80/tcp	contenedorN23
0509723d6f3f	httpd	"httpd-foreground"	56 minutes ago	Up 55 minutes	0.0.0.0:8022->80/tcp, :::8022->80/tcp	contenedorN22
84749e14fced	httpd	"httpd-foreground"	56 minutes ago	Up 55 minutes	0.0.0.0:8021->80/tcp, :::8021->80/tcp	contenedorN21
698c38d7fbc3	httpd	"httpd-foreground"	56 minutes ago	Up 56 minutes	0.0.0.0:8020->80/tcp, :::8020->80/tcp	contenedorN20
f6838a4e2cf3	httpd	"httpd-foreground"	56 minutes ago	Up 56 minutes	0.0.0.0:8019->80/tcp, :::8019->80/tcp	contenedorN19
eeF8e72a0f92	httpd	"httpd-foreground"	56 minutes ago	Up 56 minutes	0.0.0.0:8018->80/tcp, :::8018->80/tcp	contenedorN18
13b270adc526	httpd	"httpd-foreground"	56 minutes ago	Up 56 minutes	0.0.0.0:8017->80/tcp, :::8017->80/tcp	contenedorN17
bb028eadf533	httpd	"httpd-foreground"	56 minutes ago	Up 56 minutes	0.0.0.0:8016->80/tcp, :::8016->80/tcp	contenedorN16
eF0e4d350815	httpd	"httpd-foreground"	56 minutes ago	Up 56 minutes	0.0.0.0:8015->80/tcp, :::8015->80/tcp	contenedorN15
d1a99e7d2ede	httpd	"httpd-foreground"	56 minutes ago	Up 56 minutes	0.0.0.0:8014->80/tcp, :::8014->80/tcp	contenedorN14
27c34cb763d	httpd	"httpd-foreground"	56 minutes ago	Up 56 minutes	0.0.0.0:8013->80/tcp, :::8013->80/tcp	contenedorN13
81f53f4142af	httpd	"httpd-foreground"	56 minutes ago	Up 56 minutes	0.0.0.0:8012->80/tcp, :::8012->80/tcp	contenedorN12
7fe276f2271c	httpd	"httpd-foreground"	56 minutes ago	Up 56 minutes	0.0.0.0:8011->80/tcp, :::8011->80/tcp	contenedorN11
b6f2f86dab3c	httpd	"httpd-foreground"	56 minutes ago	Up 56 minutes	0.0.0.0:8006->80/tcp, :::8006->80/tcp	contenedorN6
1f19bdad9188	httpd	"httpd-foreground"	56 minutes ago	Up 56 minutes	0.0.0.0:8004->80/tcp, :::8004->80/tcp	contenedorN4
1871adb949bd	httpd	"httpd-foreground"	56 minutes ago	Up 56 minutes	0.0.0.0:8003->80/tcp, :::8003->80/tcp	contenedorN3
7fbf0876889e	httpd	"httpd-foreground"	56 minutes ago	Up 56 minutes	0.0.0.0:8002->80/tcp, :::8002->80/tcp	contenedorN2
1a2102c2fe38	httpd	"httpd-foreground"	56 minutes ago	Up 56 minutes	0.0.0.0:8001->80/tcp, :::8001->80/tcp	contenedorN1
4c4fd2d26bcd	httpd	"httpd-foreground"	59 minutes ago	Up 59 minutes	0.0.0.0:8090->80/tcp, :::8090->80/tcp	contenedorN10
43ba6c239d02	httpd	"httpd-foreground"	59 minutes ago	Up 59 minutes	0.0.0.0:8089->80/tcp, :::8089->80/tcp	contenedorN9
4cbbfa8a5e55	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8088->80/tcp, :::8088->80/tcp	contenedorN8
2261a43356e2	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8087->80/tcp, :::8087->80/tcp	contenedorN7
9940cc055f4b	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8086->80/tcp, :::8086->80/tcp	contenedorN5
711c8793549e	httpd	"httpd-foreground"	6 hours ago	Up 6 hours	0.0.0.0:8085->80/tcp, :::8085->80/tcp	contenedorEntrega
fad1205d7d18	httpd	"httpd-foreground"	6 hours ago	Up 6 hours	0.0.0.0:8084->80/tcp, :::8084->80/tcp	contenedorTrabajo
24e23e193bc1	httpd	"httpd-foreground"	7 hours ago	Up 7 hours	0.0.0.0:8083->80/tcp, :::8083->80/tcp	contenedorEjemplo
1014b072bd34	httpd	"httpd-foreground"	7 hours ago	Up 7 hours	0.0.0.0:8080->80/tcp, :::8080->80/tcp	app1

Ilustración 46

- 2) Para detenerlo ejecutamos “*docker stop*”.

```
[root@ip-10-0-24-241 ec2-user]# docker stop 4e0a81e4ec1d
4e0a81e4ec1d
```

Ilustración 47

- 3) Para visualizar los contenedores que están corriendo y el detenido ejecutamos “*docker ps -a*”.

```
[root@ip-10-0-24-241 ec2-user]# docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
4e0a81e4ec1d	httpd	"httpd-foreground"	About an hour ago	Exited (0) 3 minutes ago		contenedorN30
102fc46e02eb	httpd	"httpd-foreground"	About an hour ago	Up About an hour		contenedorN29
8727e404426	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8028->80/tcp, :::8028->80/tcp	contenedorN28
8f82690777f3	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8027->80/tcp, :::8027->80/tcp	contenedorN27
15f397a9aea9	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8026->80/tcp, :::8026->80/tcp	contenedorN26
6f5af1d9c303	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8025->80/tcp, :::8025->80/tcp	contenedorN25
652ce439cb2	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8024->80/tcp, :::8024->80/tcp	contenedorN24
24dc5a9b073b	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8023->80/tcp, :::8023->80/tcp	contenedorN23
0509723d6f3f	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8022->80/tcp, :::8022->80/tcp	contenedorN22
84749e14fced	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8021->80/tcp, :::8021->80/tcp	contenedorN21
698c38d7f3e3	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8020->80/tcp, :::8020->80/tcp	contenedorN20
f6838a4e2cf3	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8019->80/tcp, :::8019->80/tcp	contenedorN19
eef8e72a0f92	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8018->80/tcp, :::8018->80/tcp	contenedorN18
13b270adc526	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8017->80/tcp, :::8017->80/tcp	contenedorN17
bb028eadf533	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8016->80/tcp, :::8016->80/tcp	contenedorN16
efe04d350815	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8015->80/tcp, :::8015->80/tcp	contenedorN15
d1a99e7d2ede	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8014->80/tcp, :::8014->80/tcp	contenedorN14
27c34cb763d	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8013->80/tcp, :::8013->80/tcp	contenedorN13
81f53f4142af	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8012->80/tcp, :::8012->80/tcp	contenedorN12
7fe276f2271c	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8011->80/tcp, :::8011->80/tcp	contenedorN11
b6f2f86dab3c	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8006->80/tcp, :::8006->80/tcp	contenedorN6
1f19bda9188	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8004->80/tcp, :::8004->80/tcp	contenedorN4
1871adb949bd	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8003->80/tcp, :::8003->80/tcp	contenedorN3
7fbf0876889e	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8002->80/tcp, :::8002->80/tcp	contenedorN2
1a210c22fe38	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8001->80/tcp, :::8001->80/tcp	contenedorN1
4c4fd2dc6bcd	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8090->80/tcp, :::8090->80/tcp	contenedorN10
43ba6c239d02	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8089->80/tcp, :::8089->80/tcp	contenedorN9
4cbbfa8a5e55	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8088->80/tcp, :::8088->80/tcp	contenedorN8
2261a43356e2	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8087->80/tcp, :::8087->80/tcp	contenedorN7
9940cc055f4b	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8086->80/tcp, :::8086->80/tcp	contenedorN5
711c87935496	httpd	"httpd-foreground"	7 hours ago	Up 7 hours	0.0.0.0:8085->80/tcp, :::8085->80/tcp	contenedorEntre
ga						
fad1205d7d18	httpd	"httpd-foreground"	7 hours ago	Up 7 hours	0.0.0.0:8084->80/tcp, :::8084->80/tcp	contenedorTraba
jo						
24e23e193bc1	httpd	"httpd-foreground"	7 hours ago	Up 7 hours	0.0.0.0:8083->80/tcp, :::8083->80/tcp	contenedorEjemp

Ilustración 48

- 4) Si necesitamos iniciar el Docker pausado ejecutamos “*docker start*” y el id del Docker detenido.

```
[root@ip-10-0-24-241 ec2-user]# docker start 4e0a81e4ec1d
```

```
[root@ip-10-0-24-241 ec2-user]# docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
4e0a81e4ec1d	httpd	"httpd-foreground"	About an hour ago	Up 2 minutes	0.0.0.0:8030->80/tcp, :::8030->80/tcp	contenedorN30
102fc46e02eb	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8029->80/tcp, :::8029->80/tcp	contenedorN29
8727e404426	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8028->80/tcp, :::8028->80/tcp	contenedorN28
8f82690777f3	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8027->80/tcp, :::8027->80/tcp	contenedorN27
15f397a9aea9	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8026->80/tcp, :::8026->80/tcp	contenedorN26
6f5af1d9c303	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8025->80/tcp, :::8025->80/tcp	contenedorN25
652ce439cb2	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8024->80/tcp, :::8024->80/tcp	contenedorN24
24dc5a9b073b	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8023->80/tcp, :::8023->80/tcp	contenedorN23
0509723d6f3f	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8022->80/tcp, :::8022->80/tcp	contenedorN22
84749e14fced	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8021->80/tcp, :::8021->80/tcp	contenedorN21
698c38d7f3e3	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8020->80/tcp, :::8020->80/tcp	contenedorN20
f6838a4e2cf3	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8019->80/tcp, :::8019->80/tcp	contenedorN19
eef8e72a0f92	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8018->80/tcp, :::8018->80/tcp	contenedorN18
13b270adc526	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8017->80/tcp, :::8017->80/tcp	contenedorN17
bb028eadf533	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8016->80/tcp, :::8016->80/tcp	contenedorN16
efe04d350815	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8015->80/tcp, :::8015->80/tcp	contenedorN15
d1a99e7d2ede	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8014->80/tcp, :::8014->80/tcp	contenedorN14
27c34cb763d	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8013->80/tcp, :::8013->80/tcp	contenedorN13
81f53f4142af	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8012->80/tcp, :::8012->80/tcp	contenedorN12
7fe276f2271c	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8011->80/tcp, :::8011->80/tcp	contenedorN11
b6f2f86dab3c	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8006->80/tcp, :::8006->80/tcp	contenedorN6
1f19bda9188	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8004->80/tcp, :::8004->80/tcp	contenedorN4
1871adb949bd	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8003->80/tcp, :::8003->80/tcp	contenedorN3
7fbf0876889e	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8002->80/tcp, :::8002->80/tcp	contenedorN2
1a210c22fe38	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8001->80/tcp, :::8001->80/tcp	contenedorN1
4c4fd2dc6bcd	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8090->80/tcp, :::8090->80/tcp	contenedorN10
43ba6c239d02	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8089->80/tcp, :::8089->80/tcp	contenedorN9
4cbbfa8a5e55	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8088->80/tcp, :::8088->80/tcp	contenedorN8
2261a43356e2	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8087->80/tcp, :::8087->80/tcp	contenedorN7
9940cc055f4b	httpd	"httpd-foreground"	About an hour ago	Up About an hour	0.0.0.0:8086->80/tcp, :::8086->80/tcp	contenedorN5
711c87935496	httpd	"httpd-foreground"	7 hours ago	Up 7 hours	0.0.0.0:8085->80/tcp, :::8085->80/tcp	contenedorEntre
fad1205d7d18	httpd	"httpd-foreground"	7 hours ago	Up 7 hours	0.0.0.0:8084->80/tcp, :::8084->80/tcp	contenedorTrabajo
24e23e193bc1	httpd	"httpd-foreground"	7 hours ago	Up 7 hours	0.0.0.0:8083->80/tcp, :::8083->80/tcp	contenedorEjemplo
10140b72bd34	httpd	"httpd-foreground"	7 hours ago	Up 7 hours	0.0.0.0:8080->80/tcp, :::8080->80/tcp	app1

Ilustración 49

- 5) Para eliminarlo sería “*docker stop*” y el id del Docker a eliminar inicialmente para detenerlo y luego se ejecuta “*docker rm*” y el id del Docker a eliminar.

```
[root@ip-10-0-24-241 ec2-user]# docker rm 4e0a81e4ec1d
4e0a81e4ec1d
[root@ip-10-0-24-241 ec2-user]# docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
102fc46e02eb	httpd	"httpd-foreground"	2 hours ago	Up 2 hours	0.0.0.0:8029->80/tcp, :::8029->80/tcp	contenedorN29
8727ed404426	httpd	"httpd-foreground"	2 hours ago	Up 2 hours	0.0.0.0:8028->80/tcp, :::8028->80/tcp	contenedorN28
8f82690777f3	httpd	"httpd-foreground"	2 hours ago	Up 2 hours	0.0.0.0:8027->80/tcp, :::8027->80/tcp	contenedorN27
15f3f9a9aa9	httpd	"httpd-foreground"	2 hours ago	Up 2 hours	0.0.0.0:8026->80/tcp, :::8026->80/tcp	contenedorN26
6f5af1d9c303	httpd	"httpd-foreground"	2 hours ago	Up 2 hours	0.0.0.0:8025->80/tcp, :::8025->80/tcp	contenedorN25
652ce43c9cb2	httpd	"httpd-foreground"	2 hours ago	Up 2 hours	0.0.0.0:8024->80/tcp, :::8024->80/tcp	contenedorN24
24dc5a9b073b	httpd	"httpd-foreground"	2 hours ago	Up 2 hours	0.0.0.0:8023->80/tcp, :::8023->80/tcp	contenedorN23
0509723d6f3f	httpd	"httpd-foreground"	2 hours ago	Up 2 hours	0.0.0.0:8022->80/tcp, :::8022->80/tcp	contenedorN22
84749e14fced	httpd	"httpd-foreground"	2 hours ago	Up 2 hours	0.0.0.0:8021->80/tcp, :::8021->80/tcp	contenedorN21
698c38d7f8e3	httpd	"httpd-foreground"	2 hours ago	Up 2 hours	0.0.0.0:8020->80/tcp, :::8020->80/tcp	contenedorN20
f6838a4e2cf3	httpd	"httpd-foreground"	2 hours ago	Up 2 hours	0.0.0.0:8019->80/tcp, :::8019->80/tcp	contenedorN19
ee8f872a0f92	httpd	"httpd-foreground"	2 hours ago	Up 2 hours	0.0.0.0:8018->80/tcp, :::8018->80/tcp	contenedorN18
13b270adc526	httpd	"httpd-foreground"	2 hours ago	Up 2 hours	0.0.0.0:8017->80/tcp, :::8017->80/tcp	contenedorN17
bb02eadf533	httpd	"httpd-foreground"	2 hours ago	Up 2 hours	0.0.0.0:8016->80/tcp, :::8016->80/tcp	contenedorN16
ef0e4d350815	httpd	"httpd-foreground"	2 hours ago	Up 2 hours	0.0.0.0:8015->80/tcp, :::8015->80/tcp	contenedorN15
d1a99e7d2ede	httpd	"httpd-foreground"	2 hours ago	Up 2 hours	0.0.0.0:8014->80/tcp, :::8014->80/tcp	contenedorN14
27c34cbb763d	httpd	"httpd-foreground"	2 hours ago	Up 2 hours	0.0.0.0:8013->80/tcp, :::8013->80/tcp	contenedorN13
81f53f4142af	httpd	"httpd-foreground"	2 hours ago	Up 2 hours	0.0.0.0:8012->80/tcp, :::8012->80/tcp	contenedorN12
7fe276f2271c	httpd	"httpd-foreground"	2 hours ago	Up 2 hours	0.0.0.0:8011->80/tcp, :::8011->80/tcp	contenedorN11
b6f2f86dab3c	httpd	"httpd-foreground"	2 hours ago	Up 2 hours	0.0.0.0:8006->80/tcp, :::8006->80/tcp	contenedorN6
1f19bdad9188	httpd	"httpd-foreground"	2 hours ago	Up 2 hours	0.0.0.0:8004->80/tcp, :::8004->80/tcp	contenedorN4
1871adb949bd	httpd	"httpd-foreground"	2 hours ago	Up 2 hours	0.0.0.0:8003->80/tcp, :::8003->80/tcp	contenedorN3
7fbf0876889e	httpd	"httpd-foreground"	2 hours ago	Up 2 hours	0.0.0.0:8002->80/tcp, :::8002->80/tcp	contenedorN2
1a2102c2fe38	httpd	"httpd-foreground"	2 hours ago	Up 2 hours	0.0.0.0:8001->80/tcp, :::8001->80/tcp	contenedorN1
4c4fd2dc6bcd	httpd	"httpd-foreground"	2 hours ago	Up 2 hours	0.0.0.0:8090->80/tcp, :::8090->80/tcp	contenedorN10
43ba6c239d02	httpd	"httpd-foreground"	2 hours ago	Up 2 hours	0.0.0.0:8089->80/tcp, :::8089->80/tcp	contenedorN9
4cbbfa8a5e55	httpd	"httpd-foreground"	2 hours ago	Up 2 hours	0.0.0.0:8088->80/tcp, :::8088->80/tcp	contenedorN8
2261a43356e2	httpd	"httpd-foreground"	2 hours ago	Up 2 hours	0.0.0.0:8087->80/tcp, :::8087->80/tcp	contenedorN7
9940cc055f4b	httpd	"httpd-foreground"	2 hours ago	Up 2 hours	0.0.0.0:8086->80/tcp, :::8086->80/tcp	contenedorN5
711c87935496	httpd	"httpd-foreground"	7 hours ago	Up 7 hours	0.0.0.0:8085->80/tcp, :::8085->80/tcp	contenedorEntrega
fad1205d7d10	httpd	"httpd-foreground"	7 hours ago	Up 7 hours	0.0.0.0:8084->80/tcp, :::8084->80/tcp	contenedorTrabajo
24e23e193bc1	httpd	"httpd-foreground"	7 hours ago	Up 7 hours	0.0.0.0:8083->80/tcp, :::8083->80/tcp	contenedorEjemplo
1014b072bd34	httpd	"httpd-foreground"	7 hours ago	Up 7 hours	0.0.0.0:8080->80/tcp, :::8080->80/tcp	app1

Ilustración 50

De ser el caso de que deseemos hacer una eliminación forzada ejecutamos “*docker rm -f*” y el id del Docker a eliminar sin importar si está corriendo o no.

Conclusiones

- 1) **Practica de Conceptos:** Este seminario nos permitió aplicar de múltiples maneras varias practicas del conocimiento adquirido durante cada clase; conocimientos como la creación de redes virtuales (VPC), configuraciones de estancias (EC2), gestiones de grupos de seguridad y publicación de servicios en la nube de AWS.
- 2) **Ambientes mixtos:** Se logró diseñar varias arquitecturas de red con diferentes instancias (Amazon Linux y Windows Server), ejemplificando la viabilidad de los entornos mixtos en la nube y su conexión, accesibilidad y estabilidad.
- 3) **Optimización de recursos con Docker (Contenedores):** La implementación de tecnologías de contenedor nos permitió evidenciar las ventajas del manejo de virtualizaciones ligeras contra las tradicionales, demostrando su eficiencia de consumo de recursos, velocidad y portabilidad dándonos a entender la obligatoria necesidad de implementar tecnologías modernas a los proyectos.
- 4) **Fortalecimiento de habilidades:** Mejoramos conocimientos técnicos de habilidades asociadas a la administración de servidores, configuraciones de servicios web (ISS y Apache), aperturas de puertos, conexiones remotas a través de RDP y SSH, de igual forma gestionamos recursos en nube de manera segura y eficiente.
- 5) **Manejo de la nube en entornos modernos:** La experiencia de este seminario logro confirmarnos que el servicio en la nube es esencial en el desarrollo de proyectos y/u soluciones modernas que sean escalables al futuro, destacando esencialmente AWS como una plataforma robusta, pero a la misma vez versátil para el aprendizaje de proyectos reales de grandes infraestructuras.

Palabras clave

2. AWS – Plataforma de servicios en la nube
3. Servicios en la nube – Recursos tecnológicos vía Internet
4. Docker – Contenedores ligeros y portables
5. Instancia EC2 – Servidor virtual en AWS
6. Máquina virtual – Sistema operativo emulado
7. Tecnología – Herramientas para resolver problemas
8. Soluciones Modernas – Alternativas eficientes y actuales
9. Arquitectura – Estructura de sistemas tecnológicos
10. Entornos – Espacios controlados de ejecución
11. Optimización – Mejora del rendimiento tecnológico
12. Linux / Windows – Sistemas operativos
13. Comandos – Instrucciones para ejecutar acciones
14. Apache – Servidor web de código abierto
15. IP – Dirección de red única
16. Puertos – Canales de comunicación digital
17. Redes – Conexión entre dispositivos o sistemas
18. Configuraciones – Ajustes de parámetros del sistema
19. SSH – Acceso remoto seguro
20. VPC – Red virtual privada (AWS)
21. HTML – Lenguaje para páginas web
22. Sistema Operativo – Software base del computador
23. http – Protocolo de transferencia web

Referencias

1. Docker Inc. (2023). *What is a Container?*. Docker Documentation. <https://docs.docker.com/get-started/overview/>
2. Pontificia Universidad Católica de Chile. (2024). *Cómo elaborar un marco conceptual*. Recuperado de https://comunicacionacademica.uc.cl/images/recursos/espanol/escritura/recurso_en_pdf_extenso/15_Como_elaborar_un_marco_conceptual.pdf
3. Amazon Web Services. (2023). Amazon EC2 documentation. AWS Documentation. <https://docs.aws.amazon.com/ec2/>
4. IBM. (2022). What is cloud computing? IBM Cloud Learn Hub. <https://www.ibm.com/cloud/learn/cloud-computing>
5. Buyya, R., Yeo, C. S., Venugopal, S., Broberg, J., & Brandic, I. (2009). Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation Computer Systems*, 25(6), 599–616. <https://doi.org/10.1016/j.future.2008.12.001>
6. Pons, N. (2016). *Linux: principios básicos de uso del sistema*. Ediciones ENI.
7. Ortiz, C. E. E., Zavala, R. F., & Sosa, C. D. L. C. (2022). Uso de AWS Educate para un Laboratorio Virtual de Seguridad en Redes. *Revista Electrónica sobre Tecnología, Educación y Sociedad*, 9(18).
8. Patibandla, K. R. (2024). Design and Create VPC in AWS. *Journal of Artificial Intelligence General science (JAIGS) ISSN: 3006-4023*, 1(1), 273-282.