
UD1 - INTRODUCCIÓN

Contents

| | |
|---|----|
| Introducción..... | 2 |
| ¿Qué es un VPS? | 2 |
| Conexión mediante SSH..... | 3 |
| Autenticación | 4 |
| Cifrados simétricos o de clave privada | 4 |
| Cifrados asimétricos o de clave pública | 5 |
| En Clase | 6 |
| Práctica Local | 6 |
| Instalación y configuración de nuestra máquina virtual..... | 6 |
| Dar permisos de sudo a nuestro usuario | 13 |
| Configuración | 14 |
| Práctica AWS | 15 |
| Instalando AWS CLI | 19 |
| Mi primer EC2 | 20 |
| Instanciando..... | 20 |

Introducción

En este módulo vamos a simular escenarios reales donde apenas trabajaremos en local, en nuestro propio ordenador. Simularemos, mediante una máquina virtual que es en la que realmente trabajaremos, que todos nuestros despliegues ocurren en una máquina remota, tal y como ocurre en la realidad.

Además, también trabajaremos con AWS en ciertos momentos para aprender a usar la plataforma.

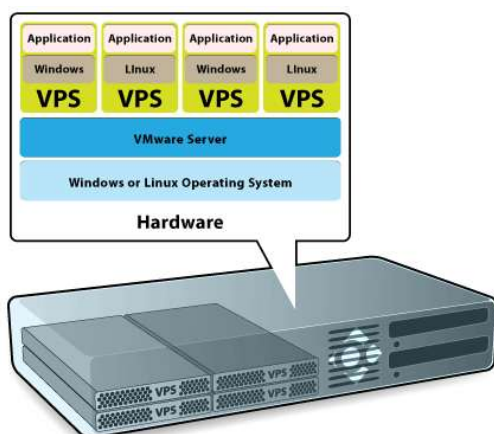
De hecho, se simulará un escenario donde tengamos contratado un VPS (Virtual Private Server) y debamos conectarnos de forma remota al mismo para poder trabajar. Un escenario muy común en el mundo real.

¿Qué es un VPS?

Un servidor es una computadora en la que tu proveedor de alojamiento web almacena los archivos y las bases de datos necesarios para tu sitio web. Cada vez que un visitante en línea quiere acceder a tu sitio web, su navegador le envía una solicitud a tu servidor y transfiere los archivos necesarios a través de Internet. El alojamiento VPS te proporciona un servidor en la nube que simula un servidor físico; sin embargo, en realidad, la máquina se comparte entre varios usuarios.

Al usar la tecnología de virtualización, tu proveedor de alojamiento web instala una capa virtual sobre el sistema operativo del servidor. Esta capa divide el servidor en particiones y le permite a cada usuario instalar su propio sistema operativo y software.

Por lo tanto, un servidor privado virtual (VPS) es tanto virtual como privado porque tienes control absoluto. Está separado de otros usuarios del servidor a nivel del sistema operativo. De hecho, la tecnología VPS es similar a la creación de particiones en tu computadora cuando quieres ejecutar más de un sistema operativo (por ejemplo, Windows y Linux) sin tener que reiniciar.



Un VPS te permite configurar tu sitio web dentro de un contenedor seguro con recursos garantizados (memoria, espacio en disco, núcleos de CPU, etc.) que no tienes que compartir con otros usuarios.

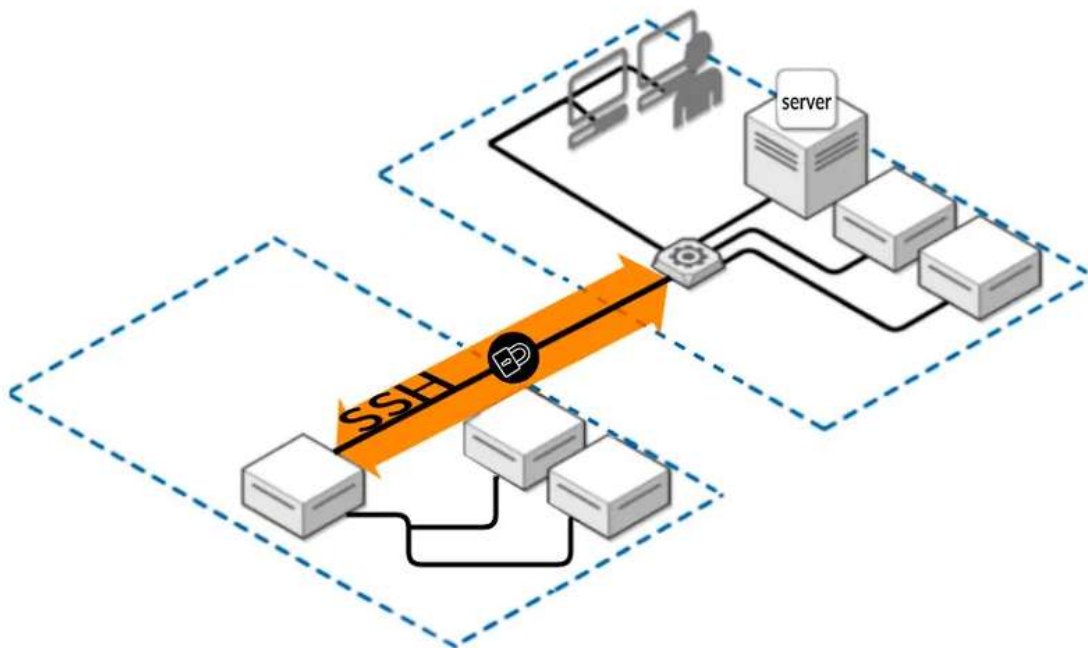
Con el hosting VPS, tienes el mismo acceso de nivel raíz que si alquilaras un servidor dedicado, pero a un costo mucho más bajo.

El VPS es una solución más segura y estable que el hosting compartido, con el que no obtienes espacio de servidor dedicado. Sin embargo, es de menor escala y más barato que alquilar un servidor completo.

El hosting VPS generalmente es elegido por los propietarios de sitios web que tienen un tráfico de nivel medio que excede los límites de los planes de hosting compartido pero que aún no necesitan los recursos de un servidor dedicado.

Conexión mediante SSH

Aunque nuestra máquina virtual esté en nuestro ordenador, ya hemos dicho que estamos simulando un VPS remoto. Para conectarnos a una máquina de forma remota y segura, la opción más recomendable es SSH.



SSH o Secure Shell es un protocolo de red criptográfico para operar servicios de red de forma segura a través de una red no protegida. Las aplicaciones típicas incluyen línea de comandos remota, inicio de sesión y ejecución de comandos remota, pero cualquier servicio de red puede protegerse con SSH.

SSH proporciona un canal seguro a través de una red no segura mediante el uso de una arquitectura cliente-servidor, conectando una aplicación cliente SSH con un servidor SSH. El puerto TCP estándar para SSH es 22 y se usa generalmente para acceder a sistemas operativos similares a Unix, pero también se puede usar en Microsoft Windows.

Proporciona un mecanismo para autenticar un usuario remoto, transferir entradas desde el cliente al host y retransmitir la salida de vuelta al cliente.

SSH tiene muchas aplicaciones diferentes:

- Gestión de servidores a los que no se puede acceder localmente
- Transferencia segura de archivos
- Creación de copias de seguridad
- Conexión entre dos ordenadores con encriptación de extremo a extremo
- Mantenimiento remoto desde otros ordenadores

Autenticación

Los dos métodos de autenticación de usuario SSH más comunes que se utilizan son las contraseñas (cifrado simétrico) y las claves SSH (cifrado asimétrico o de clave pública). Los clientes envían contraseñas cifradas al servidor de forma segura. Sin embargo, las contraseñas son un método de autenticación arriesgado porque su solidez depende de que el usuario sepa qué hace que una contraseña sea segura.

Los pares de claves pública-privada SSH encriptados asimétricamente son una mejor opción. Una vez que el cliente descifra el mensaje, el servidor le otorga acceso al sistema.

Es decir, SSH opta por el cifrado híbrido, donde se utiliza el cifrado asimétrico para intercambiar unas claves que serán las que se utilizarán posteriormente en el intercambio de información.

Este tipo de cifrado utiliza la misma clave para cifrar y para descifrar la información. Por este motivo, la clave debe ser secreta y sólo conocida por el emisor y el receptor del mensaje.

Cifrados simétricos o de clave privada

Este tipo de cifrado utiliza la misma clave para cifrar y para descifrar la información. Por este motivo, la clave debe ser secreta y sólo conocida por el emisor y el receptor del mensaje.



Ventajas

- Muy rápidos → cifrar y descifrar un mensaje cada vez requiere un cierto tiempo, que si el algoritmo es complejo, puede ser elevado.

Inconvenientes

- Si alguien no autorizado consigue la clave, podrá espiar la comunicación sin problemas
- ¿Cómo hacemos para que emisor y receptor conozcan la clave en un primer momento? → no se puede transmitir por el canal inseguro → hay que transmitirla por otro canal seguro Ejemplos: PIN de la tarjeta del banco o archivo comprimido con contraseña

Cifrados asimétricos o de clave pública

En este tipo de cifrados cada usuario utiliza un par de claves: una clave pública y una clave privada. Un mensaje cifrado con la clave pública sólo se puede descifrar con su correspondiente clave privada y viceversa.



La *clave pública* es accesible a cualquier persona que quiera consultarla, no hace falta que sea transmitida por un canal seguro como en el caso anterior.

La *clave privada* sólo la debe conocer su dueño.

Funcionamiento:

1. El emisor cifra un mensaje con la clave pública del receptor
2. El receptor recibe el mensaje y es el único que podrá descifrarlo porque es el único que posee la clave privada asociada

Ventajas

- No se necesita un nuevo canal independiente y seguro para transmitir la clave

Inconvenientes

- Son más lentos que los cifrados simétricos
- Hay que proteger muy bien la clave privada y tenerla siempre disponible para poder descifrar los mensajes (no es una contraseña)
- Hay que asegurarse de que la clave pública es de quién dice ser y no de un impostor que se esté haciendo pasar por él

En Clase

Nosotros, para conectarnos por primera vez por SSH y comprobar la conectividad, utilizaremos el cifrado simétrico (una contraseña).

Tras ello, simulando un entorno real que aporte comodidad (no introducir contraseña cada vez que hagamos *login*) pero también y sobre todo, por seguridad, utilizaremos cifrado asimétrico. Esto es, un par de claves.

Práctica Local

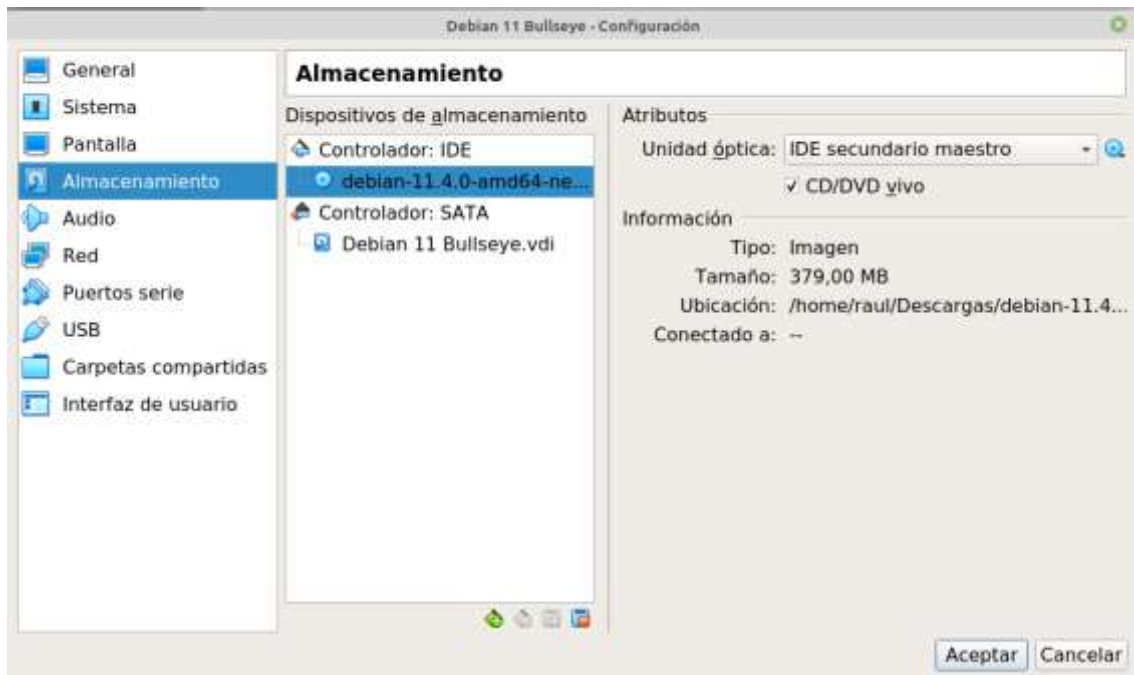
Instalación y configuración de nuestra máquina virtual

Como servidor, utilizaremos la distribución Linux Debian

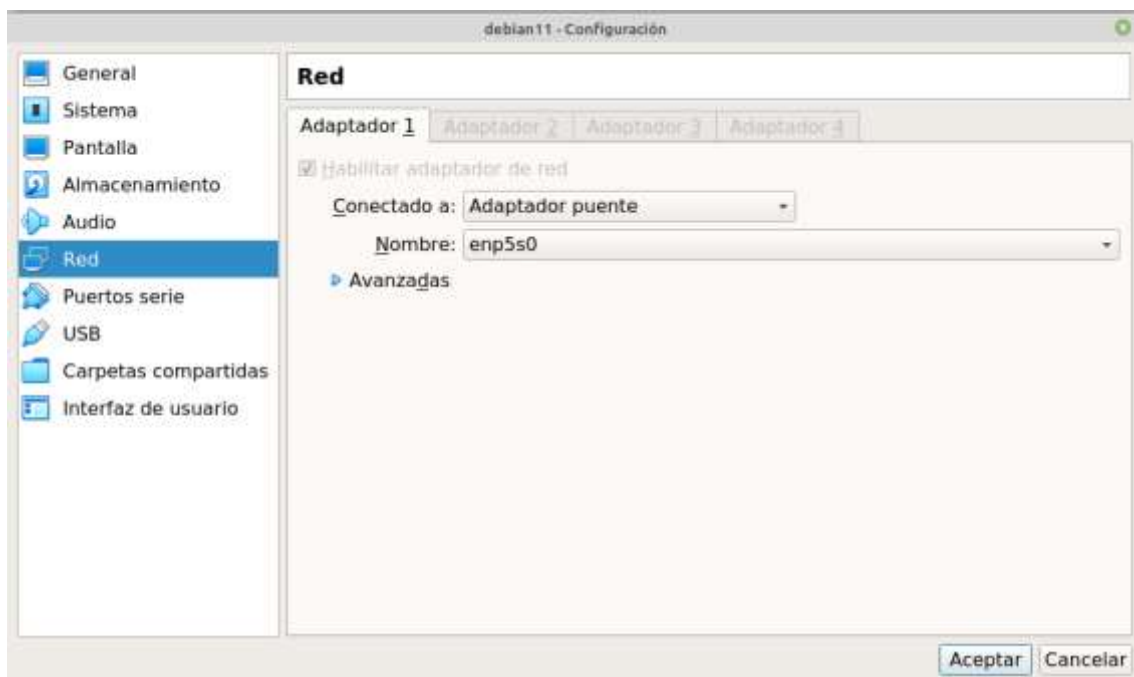
Así pues, procedamos a descargar la imagen de Debian netinstall [aquí](#)

En primer lugar, debemos crear una máquina virtual nueva, indicando su ubicación, su nombre y el tipo de sistema operativo:

Le indicamos que monte como unidad de CD la iso de netinstall de Debian que hemos descargado previamente:

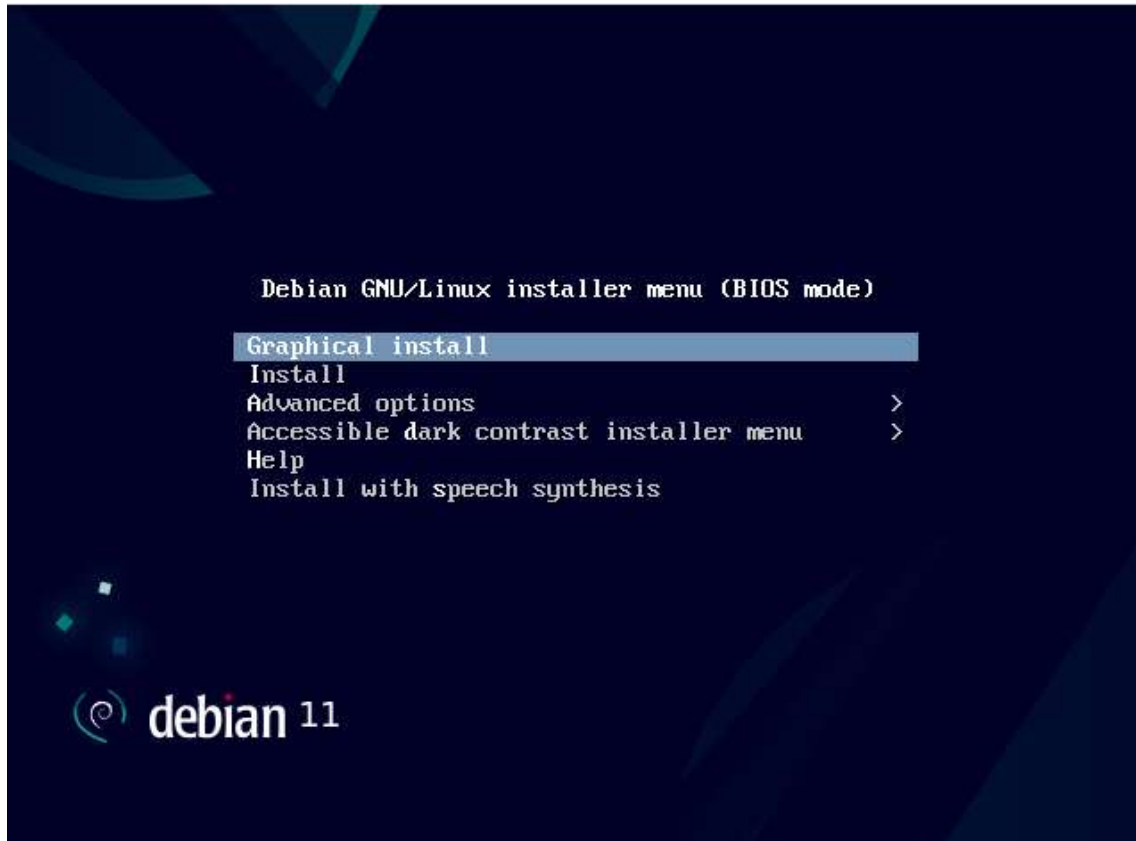


También estableceremos un único interfaz de red. Para ello, en el único adaptador de red que debe tener la máquina virtual, debemos configurarlo como tipo puente, de forma que obtenga una IP en el rango de la red local en la que nos encontremos conectados (casa, instituto...).




Sin entorno gráfico la máquina puede que funcione perfectamente con 1GB de RAM, no obstante se aconseja, si es posible, asignarle 2GB de RAM y, como mínimo, 2 procesadores.

Podéis instalar Debian tanto de forma gráfica como de forma clásica en terminal. La primera de ellas es la que os recomiendo:



Le dáis el nombre que queráis a vuestra máquina. Recomendable un nombre corto pues luego aparecerá en el *prompt* del terminal (usuario@nombredemaquina)




Configurar la red

Por favor, introduzca el nombre de la máquina.

El nombre de máquina es una sola palabra que identifica el sistema en la red. Consulte al administrador de red si no sabe qué nombre debería tener. Si está configurando una red doméstica puede inventarse este nombre.

Nombre de la máquina:

Os pedirá también contraseña de superusuario (root), nombre de vuestro usuario y contraseña para este nuevo usuario:



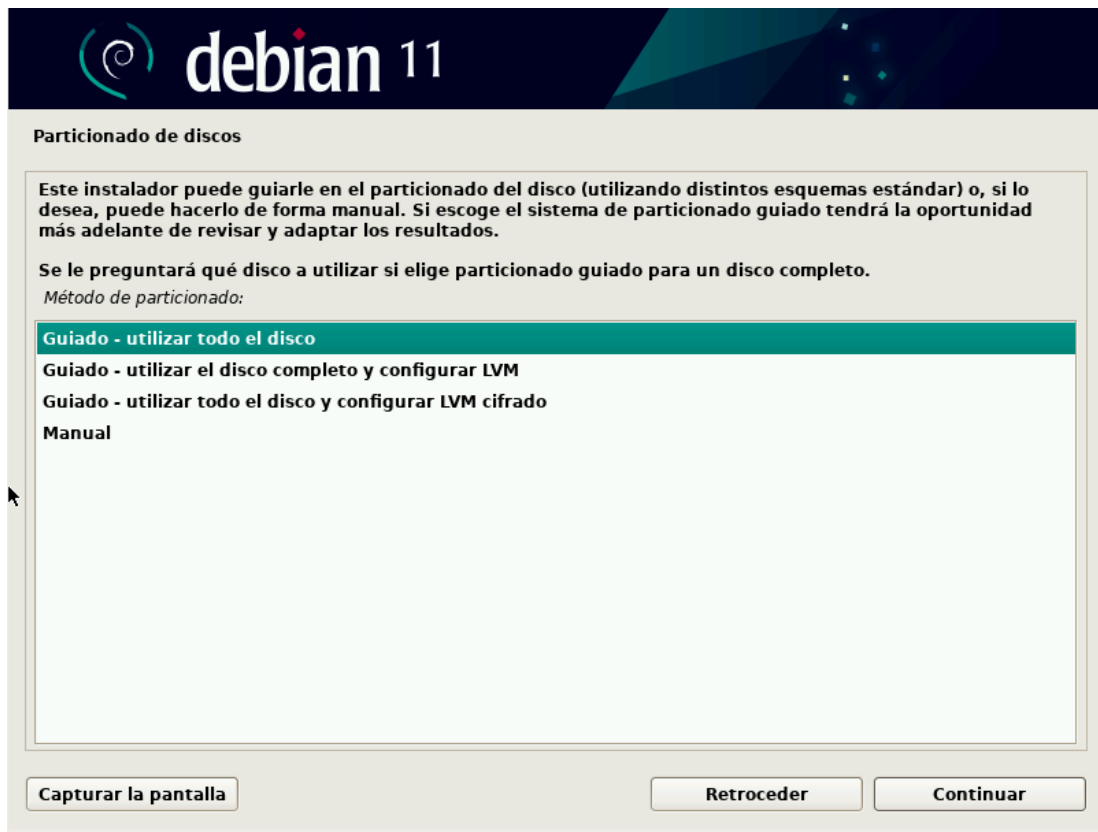
Configurar usuarios y contraseñas

Se creará una cuenta de usuario para que la use en vez de la cuenta de superusuario en sus tareas que no sean administrativas.

Por favor, introduzca el nombre real de este usuario. Esta información se usará, por ejemplo, como el origen predeterminado para los correos enviados por el usuario o como fuente de información para los programas que muestren el nombre real del usuario. Su nombre completo es una elección razonable.

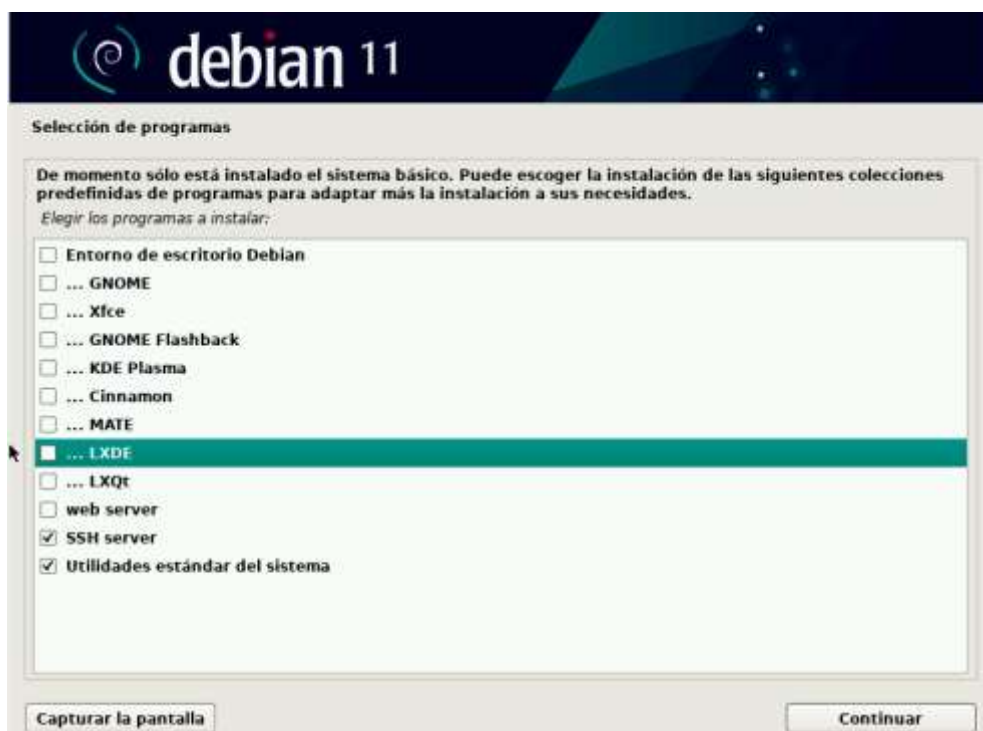
Nombre completo para el nuevo usuario:

Tras ello, para simplificar nuestro proceso, le diremos que utilice todo el disco para la instalación:



E iremos dejando todas las opciones que nos vayan apareciendo por defecto y continuando la instalación.

Tras un rato, que puede ser más o menos largo, nos mostrará la opción de instalar un entorno gráfico. En principio no nos hace falta ninguno y esta es la opción recomendada por un tema de economización de los recursos.

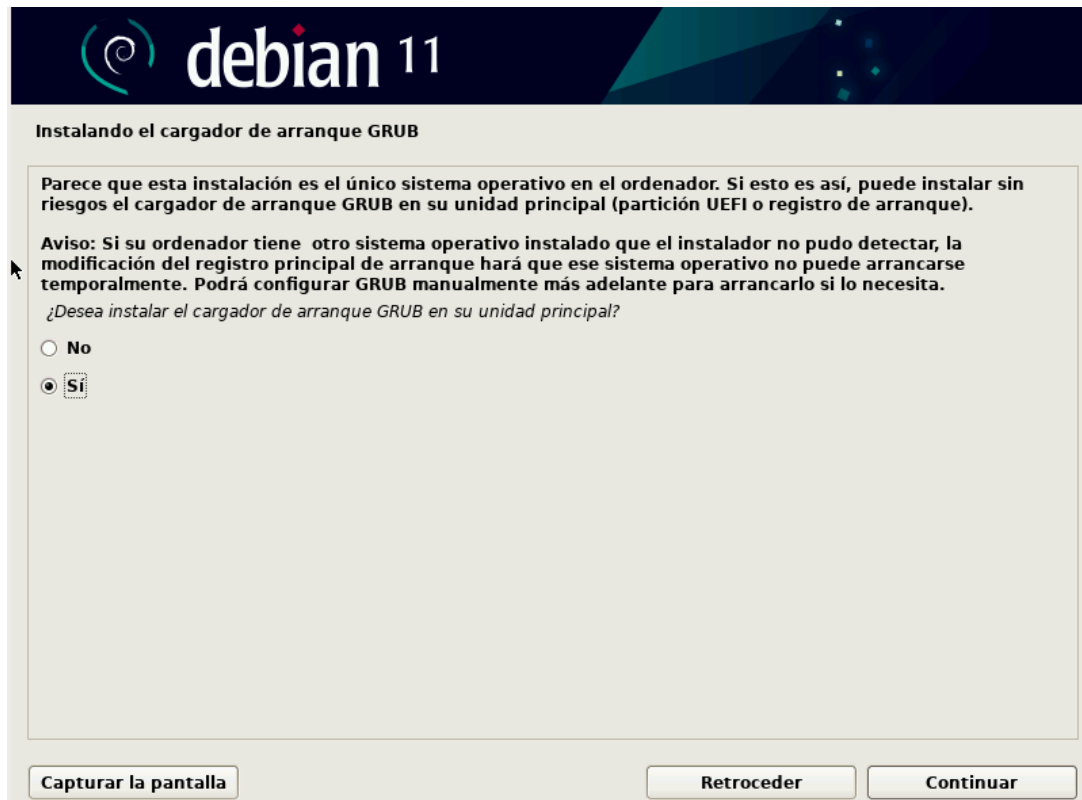


Pero si por alguna razón queréis instalar alguno, os recomiendo LXDE puesto que es el que menos recursos consume:

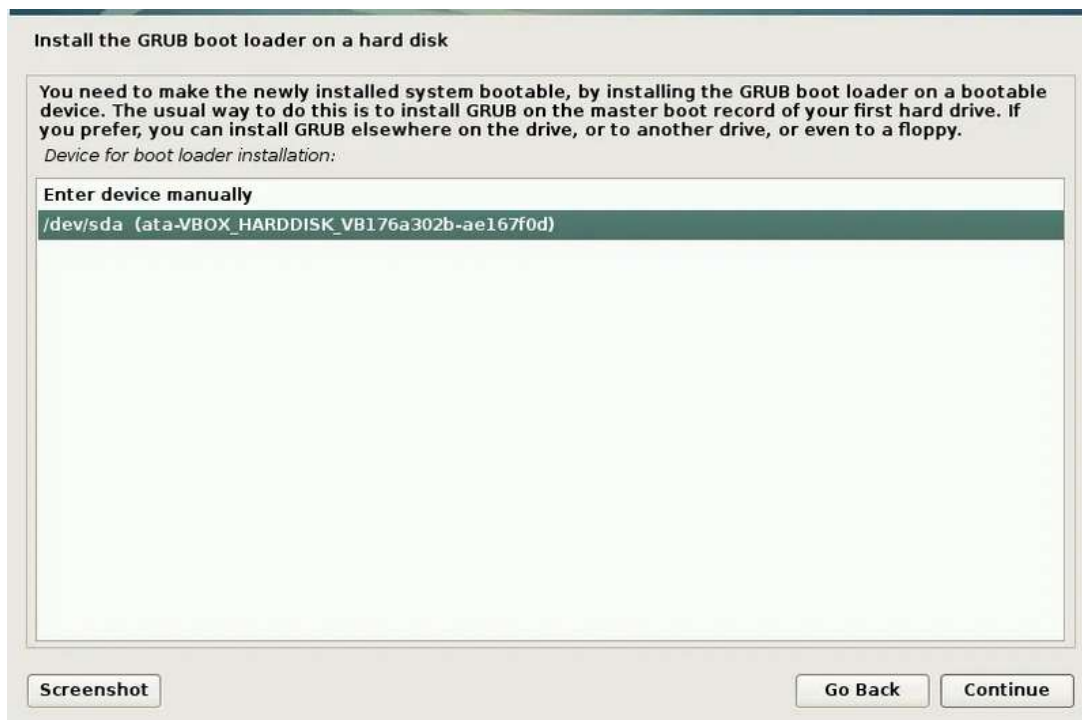


También debéis marcar las opciones que aparecen en la imagen, **SSH Server** y **Utilidades estándar**.

Le indicamos que sí que instale el gestor de arranque GRUB y continuamos con todas las opciones por defecto:



Y le indicamos que lo instale en el único disco que tenemos: /dev/sda1 (pinchad en el nombre o no lo instalará ahí)



Completará el proceso y pedirá reiniciar, cosa que haréis. Tras ello, si no tenéis entorno gráfico aparecerá un terminal pidiendo login.

Introduciendo el nombre de usuario y contraseña podremos loguearnos en el sistema.

```
Debian GNU/Linux 12 debianRobert tty1
debianRobert login: robert
Password:
Linux debianRobert 6.1.0-37-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.140-1 (2025-05-22) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
robert@debianRobert:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:86:da:53 brd ff:ff:ff:ff:ff:ff
    inet 192.168.86.41/24 brd 192.168.86.255 scope global dynamic enp0s3
        valid_lft 86302sec preferred_lft 86302sec
    inet6 fe80::a00:27ff:fe86:da53/64 scope link
        valid_lft forever preferred_lft forever
robert@debianRobert:~$ _
```

Dar permisos de sudo a nuestro usuario

Una vez instalada nuestra Debian, tendremos un usuario *raso* que es el que le dijimos que crease durante la instalación.

Puesto que a lo largo de este módulo realizaremos incontables tareas de administración, resulta un tanto incómodo, así como peligroso, el tener que cambiar de nuestro usuario a *root* cada vez que haya que instalar, configurar o modificar algo que así lo requiera.

Así pues, le daremos permisos de sudo a nuestro usuario. Estos permisos nos permitirán que cualquier comando que ejecutemos en el terminal precedido de la palabra sudo se ejecute como *root*. De la misma forma, cualquier comando que ejecutemos con nuestro usuario sin sudo, será ejecutado con los permisos de nuestro usuario, por lo que nos protegemos de *liarla* con un comando que no toca como *root*.

Dicho esto, hay varias formas de proceder, veamos la más típica y conocida. Se trata de modificar el archivo del sistema encargado de recoger estos permisos: */etc/sudoers*.

En primer lugar debemos conectarnos por SSH a nuestra máquina Debian:

```
ssh -l nombre_de_usuario IP_MV_Debian
```

Donde:

- nombre_de_usuario es vuestro nombre de usuario (el que configurastéis durante la instalación)
- IP_MV_Debian es la IP de la máquina Debian

Cambiando de nuestro usuario al usuario *root*:

```
su root
```

Preparamos el Sistema y instalamos el paquete sudo

```
apt update && apt install sudo
```

Ejecutamos la aplicación visudo que se encarga directamente de modificar el archivo de sudoers:

```
/usr/sbin/visudo
```

Y dejamos el archivo así, claro está, con vuestro propio nombre de usuario:

```
# User privilege specification
root    ALL=(ALL:ALL) ALL
robert  ALL=(ALL:ALL) ALL
```

Pulsamos CTRL+x y guardamos los cambios.

Tras esto, debemos desloguearnos de nuestra sesión SSH y volver a loguearnos. Ahora podremos validar que ya podemos realizar acciones que requieran permisos de superusuario o root. Esta validación puede realizarse con el comando:

```
sudo -v
```

Que en caso de no tener permisos nos devolverá el siguiente mensaje:

```
Sorry, user [username] may not run sudo on [hostname].
```

Y en caso de tenerlos, no devolverá nada.

Si aún así no os quedase del todo claro, podéis utilizar este comando:

```
timeout 2 sudo id && echo Access granted || echo Access denied
```

Que, en caso de tener los permisos de sudo devuelve:

```
uid=0(root) gid=0(root) grupos=0(root)
```

```
Access granted
```

Y si no los tuviera, devuelve:

```
[username] is not in the sudoers file. This incident will be reported.
```

```
Access denied
```

Configuración

En primer lugar nos crearemos nuestro par de claves, pública y privada, **en el ordenador que se conectará a nuestra debian**, con el comando (**sin sudo**):

```
ssh-keygen -b 4096
```

Si dejáis las opciones por defecto, creará una clave privada **id_rsa** y una clave pública **id_rsa.pub** en el directorio **/home/nombreusuario/.ssh**.

Os pedirá una contraseña para proteger el uso de la clave privada. Puesto que precisamente queremos agilizar el proceso de conexión por SSH para no introducir contraseñas, **debéis dejarla vacía**.

Una vez creado el par de claves, tal y como hemos visto en el apartado anterior, el servidor SSH (Debian) debe poseer nuestra clave pública para que podamos autenticarnos con nuestra clave privada, que como su nombre indica, sólo debemos poseer nosotros y por eso nos identifica unívocamente.

Este proceso de copia se puede realizar fácilmente con el comando:

```
ssh-copy-id usuario@ip_servidor
```

Al final solo estamos copiando el contenido de `id_rsa.pub` en `authorized_keys` que se puede hacer con un `echo`

```
echo "PASTE_YOUR_PUBLIC_KEY_HERE" >> ~/.ssh/authorized_keys
```

Para no tener ningún problema con los permisos sobre directorios y archivos, ejecutad en Debian:

```
chmod 700 .ssh/
```

```
chmod 600 .ssh/authorized_keys
```

Que no es más que una conexión SSH que además copia la clave, por lo que:

- usuario: nombre de vuestro usuario en Debian
- ip_servidor: ip de la máquina Debian

Práctica AWS



En vuestros correos corporativos habréis recibido un mensaje teniendo como remitente a AWS Academy.

| De | Asunto |
|--------------------------------------|---|
| <input type="checkbox"/> AWS Academy | Invitación a la asignatura Te han invitado a participar en una clase en AWS Academy . La clase s... |

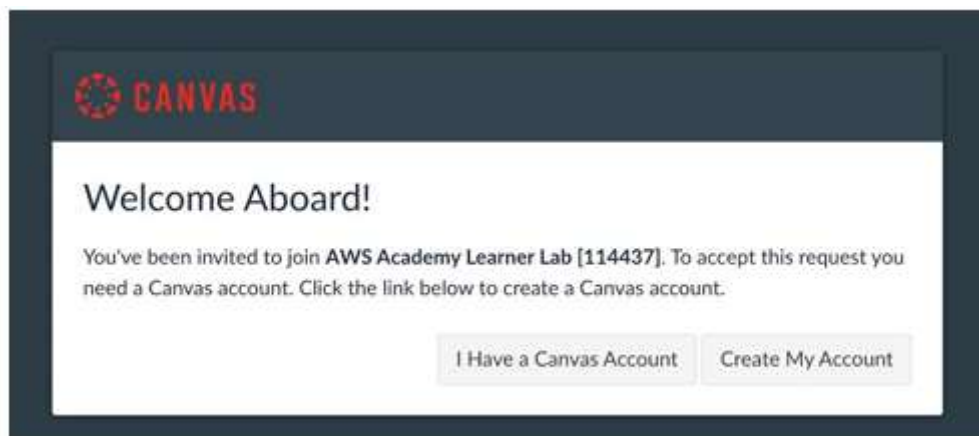
Se os ha dado de alta en un laboratorio donde vamos a ir haciendo la formación del curso, este laboratorio cuenta con un presupuesto por persona de 50\$, en función del uso que hagáis de los servicios ese dinero irá disminuyendo. El curso está preparado para que con ese presupuesto poder realizarlo en condiciones, pero también hay que indicar que es bastante común al principio dejarse

algún servicio encendido y que éste vaya consumiendo, si esto ocurriese me podéis enviar un correo indicando tal situación y ya os daría de alta en otro laboratorio.

En el correo habréis de hacer click en comenzar y registrarse en el servicio que se indica.



Para registrarse o si ya tenéis cuenta marcareis la opción más adecuada.



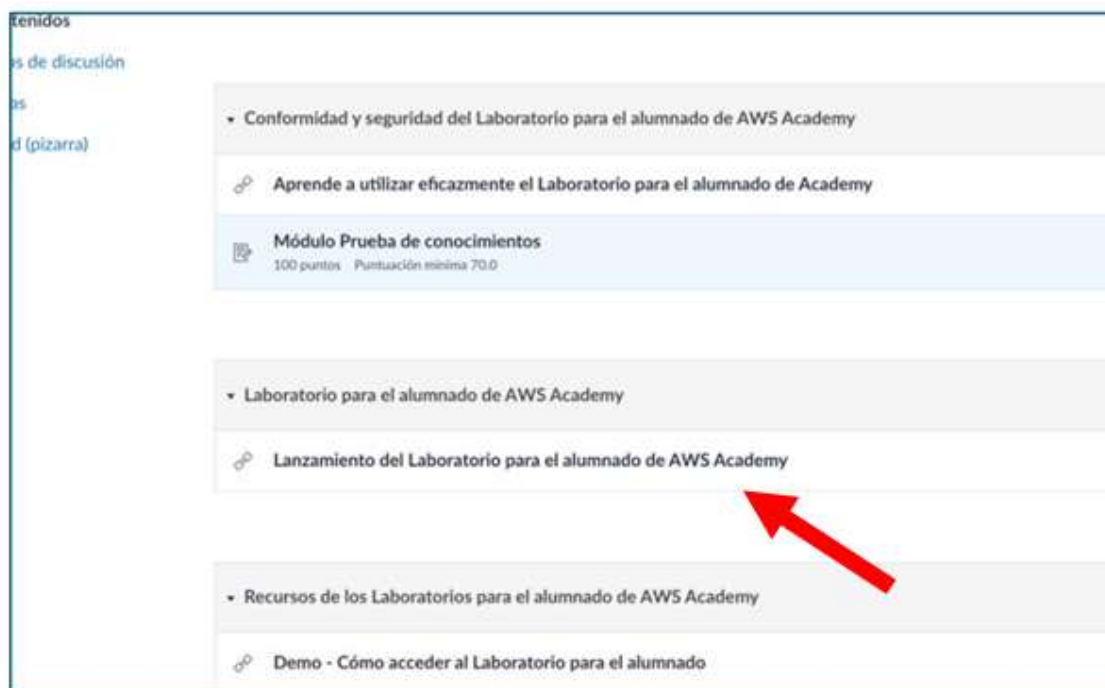
Cuando os registréis, os llevará a vuestro panel de control general



Y allí ya os mostrará el curso en el que estáis matriculados, haciendo click en él, veréis un lab estándar.



En el cual para empezar a funcionar haremos click en **Contenidos**.

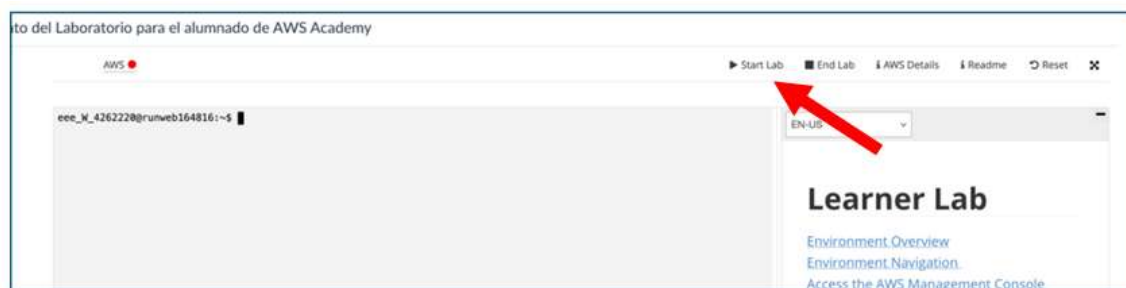


El siguiente paso será lanzar el laboratorio para poder acceder al panel de control de AWS y poder empezar a usar sus servicios. Previamente tendremos que conceder permisos y decir que nos hemos leído los términos de uso.

Una vez aceptemos se empezará a lanzar el lab.



Ahora ya tendremos casi preparado el entorno:

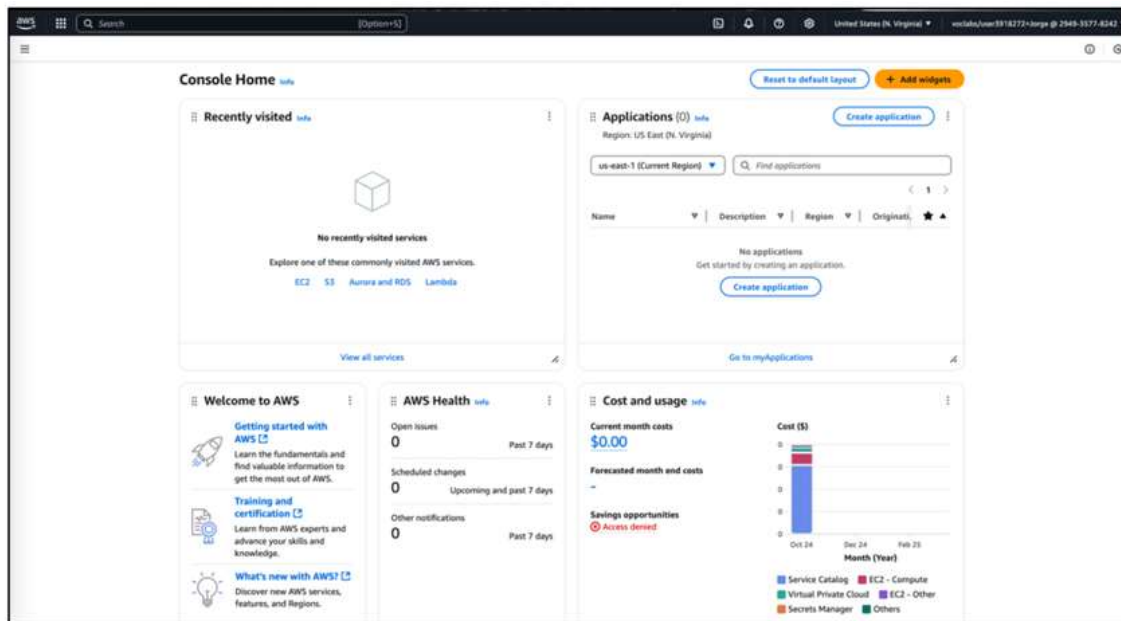


En este momento ya podremos iniciarlo, una vez iniciado tienen una duración de 4 horas de trabajo, habrá que esperar un poco ya que durante el tiempo que se inicia también está preparando las credenciales para poder conectarnos por ejemplo por ssh



Y cuando se marque en verde haciendo click en AWS podremos acceder al panel de control.





Instalando AWS CLI

AWS CLI, es el cliente de AWS mediante el cual podremos utilizar la terminal para poder trabajar con nuestro entorno en lugar de utilizar la herramienta gráfica. En este apartado indicaré la página de la documentación desde donde podremos ir siguiendo las instrucciones de instalación:

https://docs.aws.amazon.com/es_es/cli/latest/userguide/getting-started-install.html

Desde aquí podremos seleccionar el sistema operativo en el que trabajamos y poder seguir las instrucciones de instalación.



Una vez finalizada la instalación podremos comprobar la versión instalada.

```
PS C:\Users\robaa> aws --version
aws-cli/2.25.1 Python/3.12.9 Windows/11 exe/AMD64
```

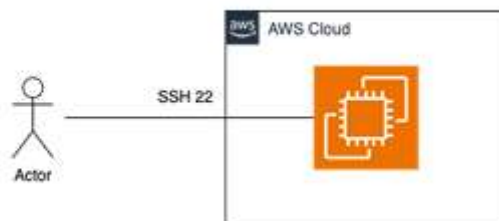
Mi primer EC2

Durante este primer tema vamos a desplegar lo que en programación se dice nuestro “hola mundo”, empezar a familiarizarnos con el panel de control y con nuestro primer servicio. En este primer tema se trata del servicio de EC2, perteneciente a la categoría de computación (al final del documento podréis encontrar un resumen de los servicios y categorías). Si buscamos la definición podríamos encontrar como tal:

“Amazon EC2 (Elastic Compute Cloud) es un servicio de computación en la nube proporcionado por AWS (Amazon Web Services) que permite a los usuarios lanzar y gestionar máquinas virtuales (instancias) de forma escalable y flexible.”

Traducido podría decirse que son las máquinas virtuales que hemos realizado hasta el momento en clase, con la única diferencia de que estos elementos de computación están expuestos en el mundo real.

A continuación, muestro el escenario con el que vamos a trabajar durante este primer tema:



En la nube de AWS es donde vamos a poder desplegar nuestros servicios. Luego éstos serán expuestos al mundo (Internet), debido a que dentro de su estableceremos las reglas de protección. Es importante recalcar que, por defecto **TODO** está denegado. Si quiero utilizar un puerto, servicio, etc ... deberé autorizarlo explícitamente.

Instanciando

Lo primero es acceder al panel de control. Si hacemos clic en la zona superior izquierda, se despliegan todas las categorías y dentro de ellas los servicios asociados. Os animo a que consultéis la gran cantidad de servicios que hay en algunas categorías.

Desplegamos la barra de búsqueda y teclearemos EC2.



Cuando busquemos el servicio EC2 podremos agregarlo a favoritos y entonces se nos añadirá a la barra de herramientas.

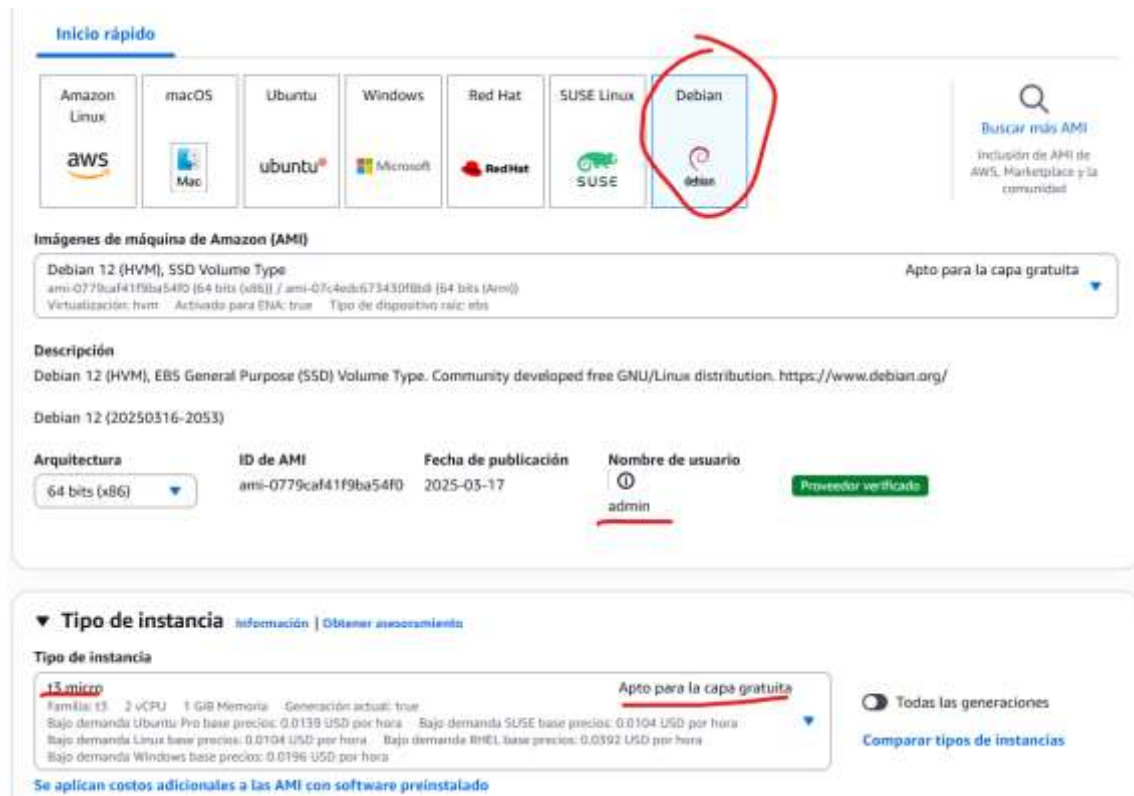
Una vez seleccionado el servicio se nos desplegará el panel de control asociado a éste.

- **Instances:** se muestran las instancias de nuestras AMIs.
- **Images:** las AMI (Amazon Machine Image) son plantillas predefinidas con un sistema operativo y software ya preconfigurado.
- **Elastic Block Store:** es el servicio de almacenamiento que proporciona almacenamiento persistente y que podemos asociar a las instancias EC2.
- **Network & Security:** aquí se definen las reglas de seguridad y un concepto como es el de IP elástica, una IP elástica es una IP pública que podemos asociar/desasociar a cualquier recurso que necesite un IP (conlleva costos asociados).
- **Load Balancing:** servicio de balanceo de carga.
- **Auto Scaling:** servicio de auto escalado. Para empezar, le daremos a Launch instance

Para empezar, le daremos a Launch instance



Lo primero es seleccionar la plantilla (AMI) en la cual basaremos nuestro EC2. Ahora seleccionaremos el Sistema Operativo y después, seleccionaremos una AMI entre las que nos ofrece del Sistema Operativo elegido. Para no gastar mucho dinero, sí, hay que estar constantemente pensando cuanto nos vamos a gastar, seleccionaremos una AMI de la capa gratuita (free tier). Aunque también podéis ir probando diferentes configuraciones para ver las ventajas y desventajas de gastar más o menos dinero.



El siguiente paso es generar una clave o reutilizar una. De forma predeterminada, en la creación del laboratorio ya tenemos el famoso vockey. Esto no es más que el .pem que encontramos en el apartado de AWS Details en la ventana de creación del lab. Este fichero .pem es la parte pública que necesita el cliente, ya que la privada se encuentra instalada en la instancia de la AMI.

¡¡Ojo que es la pantalla desde donde iniciamos el laboratorio!!



A continuación, empezamos con un aspecto fundamental, que es la parte de la configuración de seguridad. Se lleva a cabo a través de los grupos de seguridad. Un grupo de seguridad no es más que un firewall por detrás en el que, de una forma gráfica, podremos configurar las reglas de entrada y de salida.

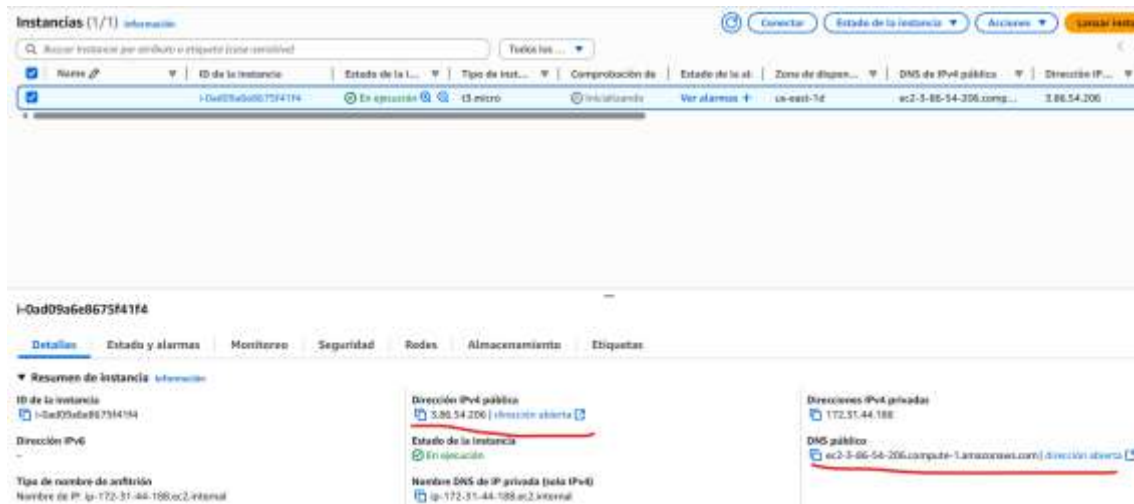
Aquí podremos ver que aparece el concepto de VPC y subred, pero eso ya lo dejaremos, ahora estamos con el “hola mundo”, y no vamos a complicarlo mucho. Al tener marcada la opción Allow SSH traffic from y luego seleccionado Anywhere, ya podremos permitir las conexiones SSH. También se puede configurar un acceso más restrictivo, especificando mejor desde dónde se puede acceder, pero eso ya lo iremos configurando cada vez más en detalle. El siguiente paso es el almacenamiento que por defecto viene con un storage de 8GB que, para lo que vamos a trabajar, es más que suficiente. A la derecha, nos aparece el número de instancias que queremos lanzar con esa configuración, por defecto viene un 1, pero para ciertas situaciones a lo mejor queremos lanzar tres o cuatro instancias a la vez. Una vez configurada ya podremos lanzar la instancia, y si todo ha ido correcto saldrá un mensaje indicándolo.

Correcto
El lanzamiento de la instancia se inició correctamente ([i-0ad09a6e8675f41f4](#))

Ahora si hacemos clic en Instances, podremos ir directamente al panel de control para ver nuestras instancias.

| Nombre | ID de la instancia | Estado de la instancia | Tipo de instancia | Comprobación de estado | Estado de la red | zona de disponibilidad | clave de clave pública | clave de clave pública | IP pública |
|---------------------|---------------------|------------------------|-------------------|------------------------|------------------------|------------------------|------------------------|------------------------|------------|
| i-0ad09a6e8675f41f4 | i-0ad09a6e8675f41f4 | En ejecución | t2.micro | Comprobación de estado | Comprobación de estado | us-east-1a | us-east-1a | us-east-1a | 10.0.0.10 |

Una vez ya tenemos la instancia en ejecución, necesitaremos ver su configuración porque vamos a probar la conexión a la misma, a ver si realmente tenemos acceso.



Recordando que habíamos descargado el archivo labuser.pem, ahora es el momento de usarlo y configurarlo. Lo primero es darle los permisos correctos.

```
chmod 400 labuser.pem
```

Y ahora ya podemos utilizarlo para conectarnos por ssh, mediante la ip pública o el nombre dns que nos ha dado.

```
PS C:\Users\robaa\Downloads> ssh -i .\labuser.pem admin@ec2-3-86-54-206.compute-1.amazonaws.com
```

Y estamos conectados

```
PS C:\Users\robaa\Downloads> ssh -i .\labuser.pem admin@ec2-3-86-54-206.compute-1.amazonaws.com
Linux ip-172-31-44-188 6.1.0-32-cloud-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.129-1 (2025-03-06) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
admin@ip-172-31-44-188:~$
```

Ahora una vez ya hemos acabado tenemos varias opciones.

- Parar el laboratorio
- Dejar que se acaben las 4 horas
- Resetear el lab. Esta opción conlleva la eliminación de todos los recursos creados.