

## AWS – Servidor de Jupyter

Para este taller vas a desplegar un servidor de Jupyter remoto en la nube de Amazon (AWS), de manera que puedas ejecutar tu código de forma remota, sin preocuparte de la capacidad y recursos de tu ordenador.

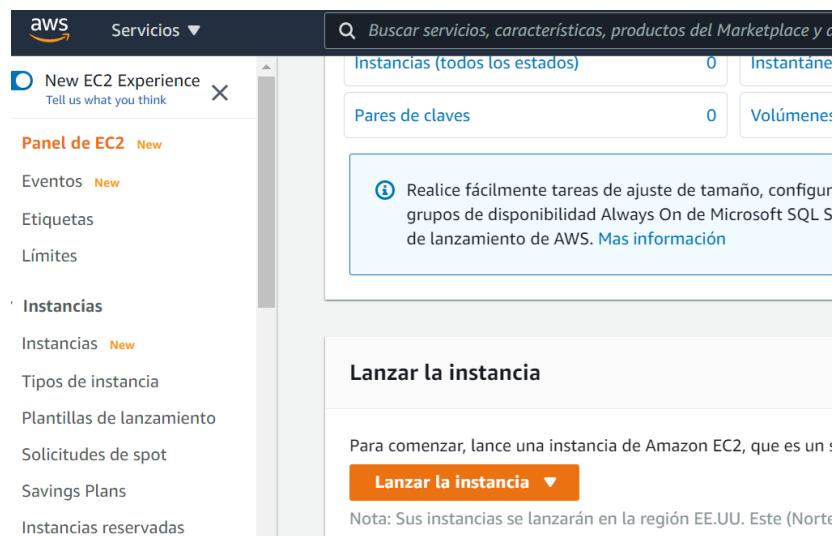
En el taller se va a crear una máquina virtual en el Cloud de Amazon (AWS), instalaremos Anaconda en la misma maquina y dejaremos lanzado Jupyter Lab. Configuraremos el acceso a la misma desde local y ejecutaremos código Python desde el navegador. Lo programamos en nuestro ordenador, pero se ejecutará en remoto, en la instancia que hemos desplegado.

### Crea una cuenta de AWS

Lo primero que necesitas es una cuenta de AWS. Puedes crear la cuenta desde [este enlace](#), además de poder consultar los servicios gratuitos del Free Tier.

### Recurso EC2

Hay que crear un recurso EC2, que es una máquina virtual en la nube de AWS. Desde el menú de arriba a la izquierda accede a Servicios -> Informática -> EC2. Una vez estemos en esta sección, seleccionamos "Lanzar una instancia":



### Configuración del EC2

Ahora tendremos que escoger una Amazon Machine Image (AMI), donde configuraremos el sistema operativo, recursos y software que vayamos a utilizar.

En "Quick Start" ya te dicen cuáles están incluidas en el Free Tier. Escogemos la primera, de momento. Es una maquina con sistema operativo Linux de Amazon.

En la pestaña de "AWS Marketplace" vienen todas las máquinas que podemos configurar. Realmente esta es una sección con templates de tipos de máquinas, con sus sistemas operativos y software necesarios para el desarrollo de nuestras aplicaciones.

Ahora dentro de este template, podremos elegir la capacidad de la misma (Memoria, Cores, SSD...). Para este taller utilizaremos la “t2.micro” del Free Tier. [Aquí tienes el detalle de los tipos de máquinas en EC2](#). No selecciones “Review and Launch”, vamos a hacer algunas configuraciones antes.

**Currently selected:** t2.micro (- ECUs, 1 vCPUs, 2.5 GHz, -)

	Family	Type	
<input type="checkbox"/>	t2	t2.nano	
<input checked="" type="checkbox"/>	t2	t2.micro	Free tier eligible

## Configuración del Security Group

Le damos a siguiente hasta llegar a la pestaña de “Security Groups”. En este taller desplegarás un sencillo servidor con jupyter, pero para algo más productivo tendrás que realizar algunas configuraciones de seguridad.

Lo primero que haremos es crear un grupo de seguridad, donde se establecerán reglas de acceso, usuarios, tráfico de entrada y salida a la máquina. Estas reglas de seguridad son independientes de esta máquina, por lo que podremos aplicar las mismas a otras que tengamos desplegadas. Reglas como:

0. Permitir protocolos HTTP, HTTPS
1. Abrir puertos

[Documentación sobre grupos de seguridad en AWS](#)

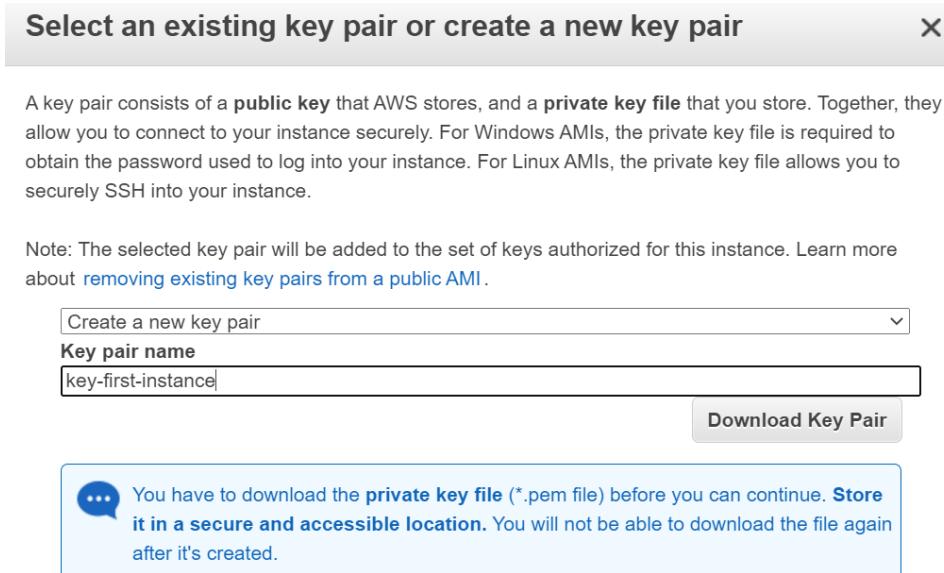
Dejamos el puerto 22 para conexiones SSH y el 8889 para TCP (“Custom TCP Rule”) para lanzar el Jupyter Lab.

Reglas de entrada (2)					<a href="#">Editar reglas de entrada</a>
Tipo	Protocolo	Intervalo de puertos	Origen	Descripción: opcional	
SSH	TCP	22	0.0.0.0/0	-	
TCP personalizado	TCP	8889	0.0.0.0/0	-	

## Claves de la maquina

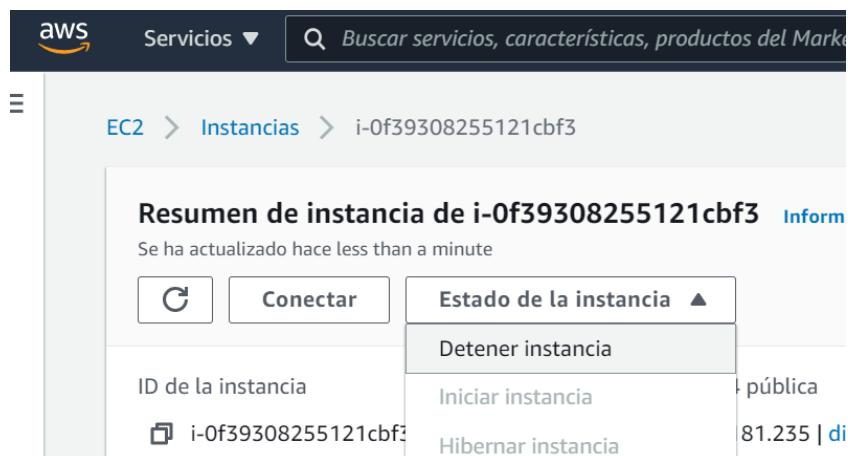
Para poder acceder a la maquina sin que nos pida las credenciales, necesitaremos una clave privada, que conservaremos en local y una clave pública que estará en el EC2. Esto se denomina “key-pair”. Esta operación se realiza al final de la configuración, tras apretar el botón del “Launch”. Creamos un nuevo key-pair **y lo utilizamos para el despliegue.**

**IMPORTANTE.** Descárgate la clave privada (archivo .pem). Es el único momento en el que podrás hacerlo. Para este taller se ha llamado “key-first-instance”, por lo que descargaremos un archivo “key-first-instance.pem”.



¡Ya tienes la máquina desplegada y corriendo! En los siguientes apartados aprenderemos a acceder a la misma, instalarle Python y dejar corriendo Jupyter.

**IMPORTANTE.** Acuérdate de apagar la máquina cuando no la estés usando. Si la dejas corriendo vas a consumir el tiempo gratuito del Free Tier.



## Acceso al EC2 desde local

Ya tenemos la máquina lista. Ahora lo que queda es acceder a la misma desde nuestro local. Para ello necesitamos conectarnos por SSH al EC2. SSH es un protocolo de comunicación entre servidores. Con este protocolo es posible introducirnos en la máquina vía terminal y realizar operaciones como crear archivos, instalar Python o correr un script.

Si estamos en Windows, abre un terminal de PowerShell, y ejecuta la siguiente sentencia. Ten en cuenta que en "key-first-instance.pem" tienes que poner toda la ruta a este archivo, "ec2-user" es el nombre de usuario que le dimos al EC2 en la creación (dejamos el que venía por defecto) y "ec2-100-25-181-235.compute-1.amazonaws.com" es el DNS público del EC2.

Podrás acceder al DNS desde la página principal de la nueva instancia.

```
ssh -i "key-first-instance.pem" ec2-user@ec2-100-25-181-235.compute-1.amazonaws.com
```

DNS de IPv4 pública  
ec2-100-25-181-235.compute-  
1.amazonaws.com | [dirección abierta](#)

Si estás en MAC, simplemente abre un terminal y ejecuta la sentencia de arriba, con los cambios comentados.

El equivalente a esto es conectarnos desde la propia web de AWS, que nos abrirá en una nueva pestaña un terminal, ya dentro de la máquina. Conectar -> Conexión de la instancia EC2 -> Conectar.

EC2 > [Instancias](#) > i-0f39308255121cbf3

Resumen de instancia de i-0f39308255121cbf3 [Información](#) [C](#) [Conectar](#)

Se ha actualizado hace less than a minute

ID de la instancia	Dirección IPv4 pública	Direcciones IP
i-0f39308255121cbf3	100.25.181.235   <a href="#">dirección abierta</a>	172.31.5

Es posible que no nos deje conectar mediante ssh dándonos un aviso de que el fichero tiene "demasiados permisos", entonces:

- Caso MAC: ejecutar `chmod 400 <nombre del fichero de claves>` (OJO estando en el mismo directorio del fichero de claves, si no, hay que poner el path del fichero)
- Caso WINDOWS:
  - 1. Abrir Power Shell
  - 2. Ir al directorio donde está el fichero de claves
  - 3. Ejecutar:
    - `icacls.exe <nombre del fichero de claves> /reset`
    - `icacls.exe <nombre del fichero de claves> /inheritance:r`
    - `icacls.exe <nombre del fichero de claves> /GRANT:R "%$env:USERNAME%:(R)"`

Esto nos permite cambiar los permisos del fichero de claves para que sólo lo pueda leer el usuario que está haciendo el ssh, que es el requisito mínimo de seguridad que exige ssh. Después de hacer lo anterior, volver a repetir el comando ssh.

Ya hemos accedido a la maquina mediante SSH. Ahora vamos a bajar e instalar Anaconda en el directorio que viene por defecto ("~/home/ec2-user/"). Esto es una maquina Linux, por lo que podrás utilizar [todos los comandos de la Shell de Unix](#).

**Para bajar el paquete de Anaconda:**

```
wget https://repo.anaconda.com/archive/Anaconda3-2018.12-Linux-x86\_64.sh
```

**Para instalar ([SI A TODO](#)):**

```
bash Anaconda3-2018.12-Linux-x86_64.sh
```

Acepta todas las condiciones de instalación.

Cambia Python2 a Python3. Por defecto, en la instancia de EC2 viene una instalación de Python 2. Vamos a cambiarla a la que acabamos de instalar:

```
source .bashrc
```

**Comprueba la versión de Python**, que debería ser la 3.X

```
python -version
```

**Lanzar jupyter**. En cuanto a acciones en el EC2, lo único que nos queda es lanzar Jupyter. Cuando abrimos Jupyter o Jupyter Lab en local, realmente se está lanzando un servicio en local, y nosotros desde el navegador mandamos peticiones al servicio. Vamos a realizar algo parecido. Lanzaremos Jupyter en el EC2, pero ahora en vez de acceder al mismo desde el propio EC2, nos conectaremos por SSH desde otra máquina (la nuestra). La interfaz será la misma, pero estará corriendo todo el código en un servidor de Amazon. **Lanzamos Jupyter**:

```
jupyter notebook --no-browser --port 8889
```

No queremos que lo lance en ningún navegador, ya que no tenemos navegador en la instancia de Amazon. Eso sí, le especificamos el puerto donde queremos que corra, ya que el que viene por defecto (8888), es el que viene por defecto....es decir, seguramente lo estemos utilizando ya en otro lado. Recuerda que los puertos nos sirven para distinguir entre diferentes servicios dentro de un mismo servidor, en este caso el servidor "ec2-100-25-181-235.compute-1.amazonaws.com".

Debería aparecer algo así al lanzarlo:

```
[ec2-user@ip-172-31-57-210 ~]$ jupyter notebook --no-browser --port 8889
[I 12:45:39.228 NotebookApp] JupyterLab extension loaded from /home/ec2-user/anaconda3/lib/python3.7/site-packages/jupyterlab
[I 12:45:39.228 NotebookApp] JupyterLab application directory is /home/ec2-user/anaconda3/share/jupyter/lab
[I 12:45:39.230 NotebookApp] Serving notebooks from local directory: /home/ec2-user
[I 12:45:39.230 NotebookApp] The Jupyter Notebook is running at:
[I 12:45:39.230 NotebookApp] http://localhost:8889/?token=1cfb54e1ef9171a83445410dde55f41413f5be0039e54e01
[I 12:45:39.230 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 12:45:39.233 NotebookApp]

To access the notebook, open this file in a browser:
  file:///run/user/1000/jupyter/nbserver-3705-open.html
Or copy and paste one of these URLs:
  http://localhost:8889/?token=1cfb54e1ef9171a83445410dde55f41413f5be0039e54e01
```

Ya tenemos el servidor corriendo, con python instalado y jupyter lanzado, esperando a ser utilizado. Queda aprovecharnos de esto de manera remota.

### Conexión desde local

Ahora vamos a acceder al Jupyter lab que corre en un servidor de Amazon, que podría estar en EEUU perfectamente, pero sin movernos de casa 😊. Para ello accederemos al EC2 creando un túnel SSH desde local. Básicamente lo que hace esto es que cuando pongamos <https://localhost:8889> en nuestro navegador (Chrome por ejemplo), nos haga un redireccionado al EC2 desplegado, puerto 8889, que es donde está corriendo el Jupyter.

Veamos cómo hacer esto. Si tu ordenador es un MAC, simplemente tendrás que abrir un terminal y ejecutar:

```
ssh -N -L localhost:8889:localhost:8889 ec2-user@ec2-100-25-181-235.compute-1.amazonaws.com -i key-first-instance.pem
```

Donde la dirección "ec2-100-25-181-235.compute-1.amazonaws.com" es el DNS público de tu EC2 y "key-first-instance.pem" es el archivo donde está la clave privada. Si no estás en el directorio donde se encuentra este archivo, tendrás que poner toda la ruta.

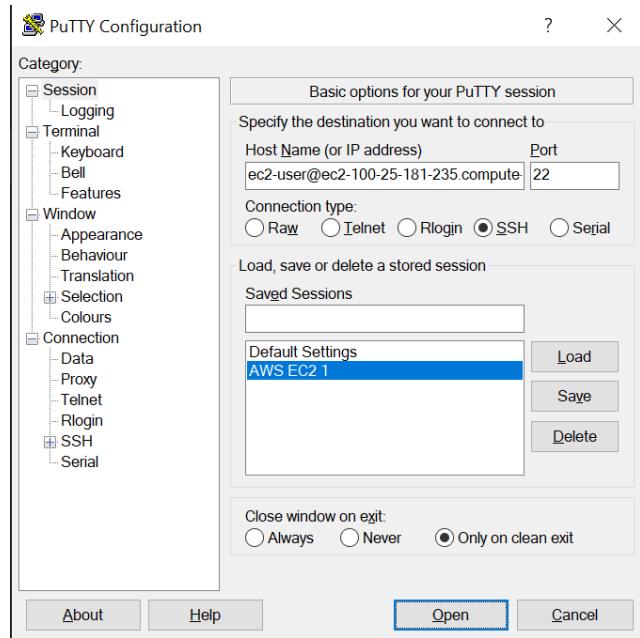
Si estás en Windows, abre un "Power Shell" y ejecuta el comando de arriba. Tiene que ser un "Power Shell" y no un "cmd", ya que en "Power Shell" tenemos habilitado el comando para conexión por SSH. Compruébalo escribiendo "ssh" en el terminal del PowerShell. Tendría que aparecer algo parecido a esto:

```
PS C:\> ssh
usage: ssh [-46AaCfGgKkMNnqsTtVvXxYy] [-B bind_interface]
           [-b bind_address] [-c cipher_spec] [-D [bind_address:]port]
           [-E log_file] [-e escape_char] [-F configfile] [-I pkcs11]
           [-i identity_file] [-J [user@]host[:port]] [-L address]
           [-l login_name] [-m mac_spec] [-O ctl_cmd] [-o option] [-p port]
           [-Q query_option] [-R address] [-S ctl_path] [-W host:port]
           [-w local_tun[:remote_tun]] destination [command]
PS C:\>
```

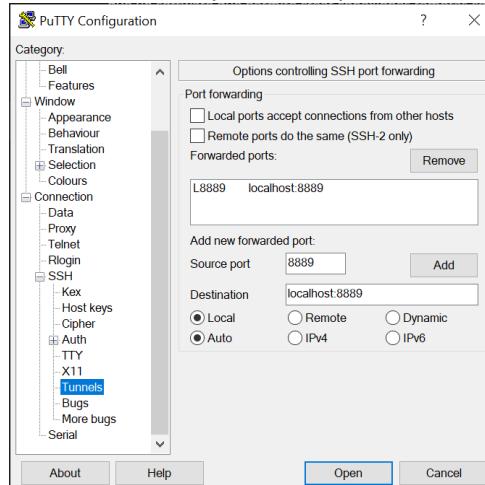
Ya estaría hecho el túnel a la máquina de Amazon.

Otra manera de hacer esto sería [desde Putty](#). Esta herramienta es muy útil y es interesante que la conozcas, aunque con lo que acabas de hacer ya estaría lista la conexión al Jupyter del EC2.

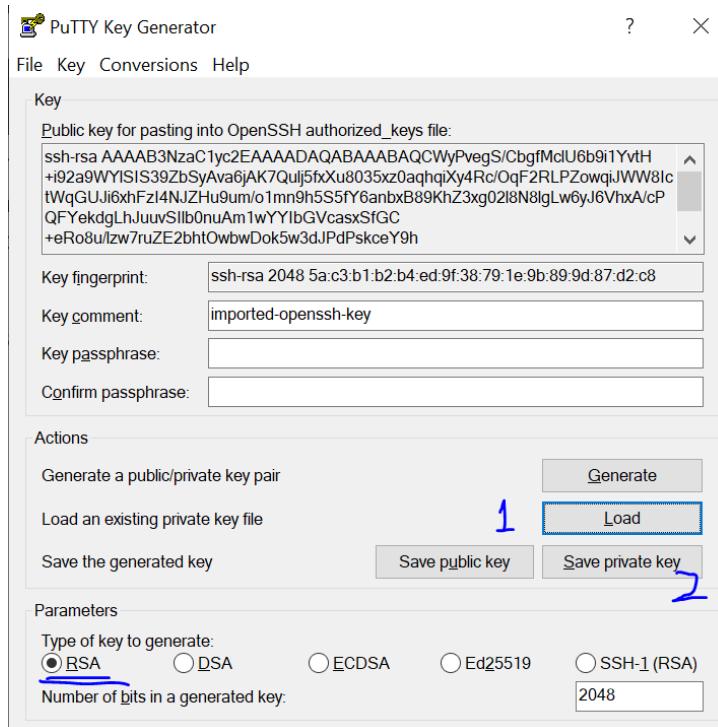
Bájate el instalador de 64 bits. No es más que un software que permite crear conexiones remotas con otras máquinas. Configura la conexión:



Y además configura el túnel (imagen de abajo). Desde el menú de la izquierda (Connection, SSH, Tunnels), pon el Destination, que será donde esté corriendo el notebook, y el Source port, que se corresponde con el puerto local donde queremos que corra.



Configura la clave privada. Putty no trabaja con los archivos “.pem” (El que nos hemos bajado de AWS). Tendremos que convertirlo a un “.ppk”. Para ello abre el PuttyGen, que se te ha descargado junto con Putty, y convierte el “.pem” a “.ppk”. Como en la imagen de abajo, asegúrate que está RSA seleccionado, después ve a “Load” y carga el archivo “.pem”. Por ultimo exportalo a un “.ppk” con “Save private key”.



Finalmente, establece la conexión (Open). Si tienes dudas sobre la conexión mediante Putty, accede a [este enlace](#).

**¡Ya lo tienes!** Ahora ve al navegador y escribe <https://localhost:8889>. Estarás accediendo al jupyter remoto gracias al tunel realizado a través de tu local.

Te va a pedir un token. Este token es el que te da el EC2 cuando dejamos corriendo el jupyter (ver abajo).

**Token authentication is enabled**

If no password has been configured, you need to open the notebook server with its login token in the UF  
paste it above. This requirement will be lifted if you [enable a password](#).

```
[ec2-user@ip-172-31-57-210 ~]$ jupyter lab --no-browser --port 8889
[I 19:56:51.539 LabApp] Writing notebook server cookie secret to /run/user/1000/jupyter/notebook_cookie_secret
[I 19:56:51.988 LabApp] JupyterLab extension loaded from /home/ec2-user/anaconda3/lib/python3.7/site-packages/jupyterlab
[I 19:56:51.989 LabApp] JupyterLab application directory is /home/ec2-user/anaconda3/share/jupyter/lab
[W 19:56:51.990 LabApp] JupyterLab server extension not enabled, manually loading...
[I 19:56:51.993 LabApp] JupyterLab extension loaded from /home/ec2-user/anaconda3/lib/python3.7/site-packages/jupyterlab
[I 19:56:51.993 LabApp] JupyterLab application directory is /home/ec2-user/anaconda3/share/jupyter/lab
[I 19:56:51.994 LabApp] Serving notebooks from local directory: /home/ec2-user
[I 19:56:51.994 LabApp] The Jupyter Notebook is running at:
[I 19:56:51.994 LabApp] http://localhost:8889/?token=0c831cd990249c6adb55f2377c250da9311dea2e80c8899b
[I 19:56:51.994 LabApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 19:56:51.997 LabApp]

To access the notebook, open this file in a browser:
  file:///run/user/1000/jupyter/nbserver-3008-open.html
  Or copy and paste one of these URLs:
    http://localhost:8889/?token=0c831cd990249c6adb55f2377c250da9311dea2e80c8899b
[In 19:56:51.997 LabApp] Cleaning up temporary cookie workspace... Localhost:8889
```

Ya tendrías un Jupyter Lab corriendo en remoto. No es una instancia muy potente en recursos pero resulta de gran utilidad para no consumir los que tenemos en el ordenador. Puedes crear notebooks, subir archivos desde el ordenador y ejecutar código.

**RECUERDA** apagar la instancia siempre que no la vayas a usar.

Cuando vayas a usarla de nuevo, la dirección pública DNS habrá cambiado. Coge la nueva dirección para lanzar la instancia y crear el túnel SSH.

