

Computación en nube con Elastic Compute Cloud

NUBARRONES

Sławomir Jastrząbski, Fotolia

Los sistemas de computación en nube como Elastic Compute Cloud (EC2) de Amazon ahorran potencia y cargas innecesarias, llevando los picos fuera de nuestros servidores. **POR DAN FROST**

Uno podría esperar que Amazon protegiera su infraestructura celosamente, pero paso a paso, la ha abierto de manera que ahora podemos meter la mano en el almacenamiento de archivos, los servidores virtuales, e incluso entregas físicas del mismo tipo (a una escala ridícula) de las que Amazon usa cada día.

Los Amazon Web Services (AWS) hacen estos sistemas disponibles sobre una plataforma de servicios Web de modo que cualquier cosa, desde más espacio en disco, crear servidores virtuales, hasta pedir entregas físicas, tienen lugar sobre el protocolo SOAP. En lugar de rellenar formularios cada vez que queramos más, menos o una infraestructura diferente, nuestro código puede permanecer tal cual y AWS proporciona los servicios según se requieran.

Cada uno de los servicios Web de Amazon incorpora herramientas desarrolladas por Amazon, y un número creciente están desarrollados por terceras compañías. Cada vez más, estas compañías están creando nuevos y complejos servicios encima de estos servicios básicos, por ejemplo, bases de datos de alta escalabilidad e indexado Web. El “Elastic Compute Cloud” de Amazon (EC2) proporciona servidores que se cobran con una tarifa por hora a partir de 10 céntimos de dólar, y que se ejecutan sobre la enorme red de servidores a lo largo de sus datacenters. EC2 nos proporciona computación en una “nube”.

El término *computación en nube* puede significar varias cosas diferentes, desde Software como Servicio (SaaS) a servicios altamente

integrables, pero también significa que no tenemos que preocuparnos por la infraestructura. EC2 se ejecuta sobre la capa de virtualización de Xen, pero no tenemos que preocuparnos por esto, sólo tenemos que pedir más servidores virtuales y éstos aparecen. La computación en nube cambia la manera en la que provisionamos servidores, ya que hace mucho más sencillo y barato el escalado rápido en los momentos de pico de demanda. En lugar de gastar varios miles de dolares en cinco máquinas que estarán el 90% del tiempo sin hacer nada, podemos usar cinco instancias EC2 sólo cuando las necesitamos.

Una de las mayores diferencias que presenta EC2 es el uso de Amazon Machine Images (AMIs), que vienen a ser configuraciones de servidor, parecidas a CDs autoarrancables. EC2 usa estas AMIs cuando crea un nuevo servidor virtual. En este artículo vamos a describir cómo crear nuestra propia AMI.

Comenzamos

Para empezar con EC2, configuramos una cuenta en AWS [1], y a continuación vamos a la página de EC2 [2], donde podemos conseguir las diferentes claves que vamos a necesitar. La manera más fácil de controlar nuestras instancias EC2 es con ElasticFox, un plugin para Firefox. Instalamos ElasticFox desde la página de Amazon Web Services [3] y echamos un vistazo. El primer paso es configurar una máquina virtual. En el centro de la ventana veremos una lista de AMIs. Para crear una nueva instancia, seleccionamos la AMI apropiada y pulsamos el botón I/O.

Un buena manera de empezar es seleccionar `ec2-public-images/fedora-core4-apache-mysql` con la ID AMI `ami-25b6534c`. La nueva instancia aparecerá abajo en la lista tras unos momentos. Cuando indique “running”, pulsamos con el botón derecho del ratón y copiamos las DNS públicas en un navegador. Ahora deberíamos ver lo que parece una página Web normal ejecutándose desde una instancia EC2.

Existe un host de AMIs disponible al público para PHP, Rails, Java, procesamiento numérico especializado y otros usos. Lo mejor de usar AMIs es que están perfeccionadas para propósitos específicos, de manera que cuando nuestra instancia EC2 está ejecutándose, no necesita ningún software innecesario. Esta configuración es de algún modo diferente al hosting tradicional, en el que el servidor suele contener todo el software para ejecutar todas nuestras aplicaciones.

Creamos una AMI

Crear una AMI lleva un rato, lo que puede parecer algo complicado al principio, pero una vez que nos ponemos, los pasos son muy sencillos. Las AMIs pueden contener cualquier cosa, desde un único servicio a todas nuestras aplicaciones y bases de datos, por lo que todas nuestras instancias EC2 serán como queramos que sean. Por ejemplo, si desplegamos muchas páginas Web, que están basadas en el mismo software, podemos colocar todo este software dentro de la AMI de manera que sólo tengamos que subirla a la instancia EC2.

Una vez que tengamos configurada una AMI, podemos usarla para crear cuantas instancias queramos. Para crear una AMI, elaboramos una imagen de Linux que contenga todos los archivos y configuraciones necesarias, empaquetamos la imagen y la subimos a EC2, para luego registrar la imagen subida.

Creamos la Imagen de Linux

El primer paso es crear la imagen de Linux en un archivo loopback, que se usa para simular un disco duro y evitar tener que crear el sistema operativo en otra unidad. El comando `dd` copia la información en bruto en un archivo de tamaño especificado, en este caso, 1GB:

```
dd if=/dev/zero ➤
of=myimage.fs count=1024 ➤
bs=1M
```

El programa `dd` trabaja con bloques como unidad de medida. `count` es el número de bloques que se va a copiar y `bs` es el tamaño de

los bloques a usar. Al ejecutar esto se genera un archivo totalmente vacío, dentro del cual se creará la imagen de Linux.

A continuación tenemos que crear un sistema de archivos con *mke2fs*, el cual añade un sistema ext3 en el archivo que acabamos de crear:

```
mke2fs -F -j myimage.fs
```

Si no hemos creado sistemas de archivo en dispositivos loopback con anterioridad, este paso puede parecer un poco raro. Podemos considerarlo de la siguiente manera: el archivo *myimage.fs_* contiene ahora un sistema de archivos que puede montarse de la misma forma que un disco duro externo:

```
sudo mount -o loop ➤
myimage.fs /mnt
```

Este paso monta el sistema de archivos en */mnt*. La opción *-o loop* hace que se monte el sistema de archivos como loopback, en lugar de como un disco duro real. El dispositivo loopback completo debería empezar a colocarse en su sitio en este momento. Si echamos un vistazo a */mnt* lo que veremos de momento es el habitual directorio *lost + found* de ext3. Ahora ya podemos crear archivos y directorios: este sistema de archivos contendrá una versión reducida de Linux para usar con EC2.

Para comenzar con nuestro sistema operativo básico podemos usar *debootstrap*, un programa que configura un sistema Debian básico en un sistema de archivos dado. En caso de que necesitemos instalar este programa, sólo hace falta teclear *apt-get install debootstrap*. Si estamos haciendo esto desde otra distribución de Linux, los pasos son similares, aunque puede que tengamos que configurar el sistema operativo con alguna diferencia. Pero para nuestro ejemplo, ejecutamos *debootstrap*:

```
sudo debootstrap --arch ➤
i386 edgy mnt
```

Mientras que se ejecuta *debootstrap*, veremos varios mensajes de recuperación y validación conforme obtiene los archivos necesarios y los instala en el sistema de archivos loopback. Cuando finalice el programa, podemos echar otro vistazo a */mnt*. Ahora tendrá un aspecto familiar. Con los siguientes comandos, salimos de esta tarea y nos ubicamos en la nueva imagen de Linux como si se hubiera arrancado:

```
sudo cp /etc/apt/sources.list ➤
/mnt/etc/apt/sources.list
sudo chroot /mnt
mount -t proc none proc
```

Ahora que ya estamos en la imagen, cambiamos la contraseña:

```
passwd
```

Lo que tenemos ahora es una imagen vacía de Debian, que tiene un uso bastante limitado. Empleamos *Aptitude* para actualizar la imagen e instalar un servidor SSH:

```
aptitude update
aptitude upgrade
aptitude install openssh-server
```

Si queremos Apache, tecleamos:

```
aptitude install apache2
```

A continuación deberíamos arreglar las configuraciones de red. Nótese que editamos */etc/network/interfaces* de la imagen, pues ya no está en nuestra máquina local. Usamos un editor para añadir lo siguiente a */etc/network/interfaces*

```
auto lo
iface lo inet loopback
auto eth0
iface eth0 inet dhcp
```

y luego añadimos lo siguiente a */etc/fstab*:

```
/dev/sda2 /mnt ext3 ➤
defaults 1 2
/dev/sda3 swap swap ➤
defaults 0 0
```

Ahora ya tenemos la imagen, y podemos probarla un poco e incluso quizás instalar algún software (por ejemplo, *subversion*, *MySQL* o cualquier otra cosa que usemos con frecuencia). Una vez que hayamos finalizado, tecleamos:

```
exit
sudo umount /mnt
```

Ahora tenemos Linux en un archivo. A estas alturas, ya podemos imaginar cómo usa EC2 esto para crear instancias. Cualquier programa, código o archivos necesarios por nuestras aplicaciones pueden añadirse a la nueva imagen de Linux en esta etapa. Todo lo que añadamos ahora estará

en cada una de las instancias de EC2 que creemos. Por ejemplo, podemos copiar archivos directamente en la imagen montada. Si usamos algún software CMS estándar, podríamos tomar una copia para la imagen: *svn co http://svn.server/my_project/trunk mnt/var/www/html/my_project*.

Ahora podemos configurar bases de datos, enlaces simbólicos, archivos de configuración o cualquier otra cosa que haríamos normalmente. Una vez hayamos terminado con esta configuración, ya tenemos lista la imagen de Linux.

Preparar y Subir

Amazon proporciona dos juegos de herramientas. El primer conjunto de software que necesitamos es el paquete *AMI Tools* [4], que contiene las herramientas para crear AMIs y subirlas a Amazon. El segundo es el paquete de herramientas en línea de comandos para EC2 [5], que permite llevar a cabo tareas más genéricas, como crear y controlar instancias EC2. Para comenzar, descargamos ambos archivos y los desempaquetamos en un directorio. Aunque podemos instalarlos en un directorio del sistema (*/usr/local* por ejemplo), para este ejemplo, los vamos a instalar en el directorio de usuario. Con los archivos en su sitio, configuramos algunas variables de entorno. El software EC2 requiere un par de variables a medida:

```
export EC2_HOME=➤
~/ec2-api-tools/
export EC2_AMITOOL_HOME=➤
~/ec2-ami-tools/
```

Comprender las Claves

EC2 presenta toda una gama de claves, IDs de acceso, certificados y demás:

- Clave Id de acceso, una cadena de 20 caracteres que identifica todas las peticiones que hagamos a los Amazon Web Services.
- Clave secreta de Acceso, una cadena de 40 caracteres que valida la clave de acceso.
- Archivo de certificado X.509, una clave pública y una clave privada.
- Archivo de clave privada.

KeyPairs nos conecta a instancias sin necesidad de enviar nuestra contraseña en texto claro. Para nuestros ejemplos, los generamos con *ElasticBox*.

Para más información acerca de estas variables, podemos leer el archivo `Readme ec2-ami-tools/readme-install.txt`.

A continuación nos aseguramos de que `JAVA_HOME` está activada y añadimos los directorios EC2 a la variable `PATH`.

```
export JAVA_HOME=
/usr/lib/jvm/cacao/jre/
export PATH=$PATH:
ec2-api-tools/bin:
ec2-ami-tools/bin
```

Para verificar que todo funciona, tecleamos:

```
ec2-bundle-image -help
```

Para usar nuestra imagen Linux, necesitamos empaquetarla, subirla a EC2 y registrarla. Para empaquetar la imagen, usamos la herramienta `ec2-bundle-image`, proporcionada por las AMI tools:

```
ec2-bundle-image
-i myimage.fs
-cert ec2-keys/cert-XXX.pem
-privatekey ec2-keys/
pk-XXX.pem
-u 1234-2345-1234
```

Esto toma nuestra imagen de Linux, la divide en varios archivos, y crea un archivo de manifiesto que indica a EC2 en qué lugar de los Amazon Simple Storage Services (S3) se ubica y cómo usarlo. Los archivos de imagen divididos se crean en `/tmp` por defecto: podemos echarle un vistazo una vez que se complete el proceso `ec2-bundle-image`.

Seguidamente, subimos la imagen con la herramienta `ec2-upload-bundle`, que toma todos los archivos que acabamos de crear en nuestra máquina local y los subimos a S3:

```
ec2-upload-bundle
-b my-image \
-m /tmp/myimage.
fs.manifest.xml
-a access-key-here
-s secret-key-here
-ec2cert ~/test1518.pem
```

Esto puede llevar algo de tiempo, por lo que debemos asegurarnos de que nuestra terminal no hace timeout mientras esperamos (por ejemplo, con `screen`). Una vez completada la subida, miramos en nuestra cuenta S3 y veremos que el contenedor con el nombre `my-image` contiene los archivos que hemos creado con `ec2-bundle-image`.

Nuestra imagen de Linux está ahora en S3 con su archivo manifiesto.

Registrar y Usar la AMI

El último paso es registrar y usar la imagen de Linux:

```
ec2-register
my-ubuntu-df/
myimage.fs.manifest.xml
-K ~/.ec2/pk-XXXX.pem
-C ~/.ec2/cert-XXXX.pem
```

Nótese que `ec2-register` refiere al archivo de manifiesto en S3, no al de nuestra máquina local, de ahí la ruta `my-ubuntu-df/myimage.fs.manifest.xml`. De igual modo, podemos registrarla mediante ElasticBox con sólo pulsar el icono verde del signo positivo en el listado de AMI y teclear la ruta en el archivo de manifiesto.

Para usar la imagen, abrimos ElasticBox, refrescamos la lista de AMIs y encontramos nuestra nueva AMI con el cuadro de búsqueda arriba a la derecha de la lista de AMIs. Creamos una nueva instancia de la AMI, y ahí está: ya estamos ejecutando nuestra propia imagen de Linux en EC2 por 10 céntimos de dólar a la hora.

Una vez que la instancia se esté ejecutando, hacemos `ssh` sobre ésta para poder jugar un poco. Tan pronto como decidamos qué software y archivos queremos en todas nuestras instancias de EC2, podemos colocar los archivos y programas en la AMI con los pasos que hemos visto.

Si pensamos que nuestra imagen es muy buena, podemos compartirla con otros gratuitamente o bien cobrando por su uso sobre Amazon.

Jugando con las Nubes

Como toda nueva tecnología, la computación en nube es divertida para experimentar y jugar un rato, pero lo mejor sería buscarle un uso adecuado.

Pero... ¿para qué es bueno EC2? La computación en nube facilita liberarse de grandes cantidades de hardware sin tener que preocuparnos de los detalles del hosting, conectividad de red, refrigeración o el aburrimiento de enfrentarse a 100 contratos de hosting. Esto hace que EC2 sea bueno para todo aquello que requiera grandes cantidades de servidores: procesamiento de millones de imágenes, tareas de búsqueda y catalogación, etc. Todo lo que pueda hacerse más rápido poniendo más capacidad de cómputo, puede usarse con EC2.

Y debido a que podemos requerir servidores al vuelo, la computación en nube es buena para tareas críticas con el tiempo, como enviar cientos de elementos por email a la hora de comer o preparar grandes cantidades de archivos de vídeo con los usuarios esperando. El escalado al vuelo significa que no tendremos docenas de servidores ociosos (y costando un buen dinero).

La computación en nube también es adecuada para cualquier servicio que necesite escalarse, pero del que no sabemos el número de usuarios finales, por ejemplo, redes sociales, intranets, extranets o aplicaciones en línea. También podemos usar EC2 para probar nuevas configuraciones de servidor, y podemos usar la nube para probar las aplicaciones [6].

La computación en nube ha nacido para cambiar la manera en que se construyen y distribuyen las aplicaciones. Todo lo que era imposible hasta hoy por falta de presupuesto en los servidores se ha vuelto súbitamente posible, o al menos mucho más barato. El poder crear AMIs a medida nos permitirá sacar el máximo provecho del servicio lanzando instancias EC2 ajustadas a nuestras aplicaciones particulares. El proceso de crear y subir las imágenes puede llevar tiempo, pero una vez que las tenemos, es fácil modificarlas para que contengan exactamente las aplicaciones que necesitamos y muchas más.

Una vez que podamos crear 1.000 copias de nuestra aplicación, ya podemos dejar de preocuparnos acerca de "cargas de servidor". ■

RECURSOS

- [1] Crear una cuenta AWS: <https://aws-portal.amazon.com/gp/aws/developer/registration/index.html>
- [2] Página de EC2: <http://www.amazon.com/ec2/>
- [3] Amazon Web Services: <http://developer.amazonwebservices.com/connect/entry.jspa?entryID=609>
- [4] Herramientas AMI de EC2: <http://developer.amazonwebservices.com/connect/entry.jspa?externalID=368&categoryID=88>
- [5] Herramientas en línea de comandos de Amazon: <http://developer.amazonwebservices.com/connect/entry.jspa?externalID=351&categoryID=88>
- [6] Selenium: <http://selenium-grid.openqa.org/>