El cloud

1. Principio

El Cloud, o Cloud Computing, o computación en la nube, consiste en el uso de medios informáticos, generalmente servidores, remotos o a través de una red. En el marco de un Cloud público, la red sería Internet. La palabra cloud, o nube, indica que el usuario no sabe, o no necesita saber, la ubicación de los servidores.

El cloud también puede ser privado. En este caso, la infraestructura asociada queda gestionada por una empresa por cuenta propia, aunque propondrá al conjunto de servicios o entidades medios idénticos a los de un cloud público: servidores, redes, publicaciones, etc.

Las diferencias entre un cloud y un servicio clásico son:

- Recursos de acceso libre: por ejemplo crearemos, usaremos, modificaremos o simplemente eliminaremos un servidor según sus necesidades y medios, generalmente, de manera automática, con una respuesta inmediata.
- El pago por uso o Pay-per-use: solamente se pagará lo que se consuma. El consumo lo mide el operador, no solamente para adaptar su capacidad (añadido de recursos físicos o lógicos), sino también para la facturación.
- La mutualización: las infraestructuras físicas (servidores, discos, redes) son compartidas por el conjunto de los clientes y pueden ser heterogéneas. Las capas superiores están completamente virtualizadas: máquinas virtuales, contenedores, redes, repartidores de carga, etc.
- La apertura: ya sea un cloud público o privado, el usuario accede a los servicios a través de terminales web, API, CLI, etc. Esto permite un alto nivel de automatización.

Una palabra clave es la **elasticidad**. Se dice que el cloud es elástico, ya que es capaz de adaptarse más rápidamente a las necesidades expandiendo o reduciendo sus servicios bajo demanda. También se habla de plasticidad y escalabilidad.

Sin embargo no hay que olvidar que, detrás del cloud, hay servidores de verdad, en

datacenters de verdad, con personas de verdad que los administran. No hay nada mágico. Es posible que haya algún momento en el que no pueda crear una nueva instancia (servidor virtual) y que reciba un mensaje del operador diciéndolo que no tiene recursos disponibles, o incluso anunciando una avería, o la pérdida de uno o varios servicios en una región o zona de disponibilidad.

Nos concentraremos aquí en el cloud público.

2. Servicios cloud

Se distinguen distintos servicios propuestos por los operadores, reconocibles gracias al sufijo "aaS" que significa "as a Service". Todo es servicio en el cloud, he aquí los principales:

- **IaaS**, *Infrastructure as a Service*. Es la virtualización de un parque informático: el operador propone servidores virtuales y los servicios asociados, principalmente la red y el almacenamiento. El usuario no tiene que preocuparse de la infraestructura física sobre la que reposan los servicios, sino que se concentra únicamente en la definición de su infraestructura virtual.
- **PaaS**, *Platform as a Service*. La infraestructura (servidores, redes, OS, etc.), es administrada por el operador. El usuario dispone de una plataforma llave en mano donde puede desplegar y controlar directamente sus aplicaciones y administrar las interacciones entre los distintos componentes aplicativos, normalmente se trata de contenedores.
- **SaaS**, *Software as a Service*. En esta abstracción, el operador proporciona las aplicaciones. Si en el PaaS el usuario despliega su servidor web (apache, por ejemplo) él mismo, o su sistema de gestión de base de datos (MySQL por ejemplo), en el SaaS, se proporcionan los componentes, las actualizaciones, los parches de seguridad, e incluso a veces las migraciones.

Estos servicios son apilables: un PaaS se puede instalar en un IaaS. Un SaaS se puede instalar en un PaaS o en un IaaS. Un ejemplo de PaaS es OpenShift, de Red Hat, que se instala obligatoriamente en un IaaS. Sur OpenShift, se puede instalar y dar acceso a aplicaciones en modo SaaS.

Esta lista no es exhaustiva. De ella derivan decenas de ofertas de servicios: NaaS, *Network as a Service*, STaaS, *Storage as a Service*, CaaS, *Communication as a Service*, DaaS, *Desktop as a Service*, *Data as a Service*, y otras ofertas.

3. Proveedores

Los grandes proveedores de soluciones de cloud público son Amazon, Microsoft y Google. Amazon es el precursor, por eso nos vamos a concentrar en los servicios que propone, los otros proponen servicios equivalentes o parecidos.

4. Ejemplo de AWS

Amazon lanzó en 2002 un servicio de Cloud llamado **AWS**, *Amazon Web Services*, para cubrir sus necesidades internas. Al final del año 2003, ya pensaba en una apertura al público. Un primer servicio se hizo público en 2004, pero fue en marzo de 2006 cuando Amazon abrió públicamente su IaaS.

De entre las decenas de servicios propuestos por AWS, se distinguen de forma no exhaustiva:

- **EC2**: *Elastic Compute Cloud*: el servicio permite alquilar servidores, con toda su infraestructura. Se elige el tipo de instancia (CPU, memoria) en función de la necesidad, el almacenaje (discos clásicos o SSD), la red, etc. Los servidores pueden llevar una dirección IP pública, o una dirección IP privada de un VPC.
- **AMI**, *Amazon Machine Image* es una imagen de servidor necesaria para la instalación y el arranque de una instancia EC2. De hecho son instantáneas (snapshots) de volúmenes EBS, que contienen un sistema operativo o aplicaciones.
- **S3**: *Simple Storage Service*, para el alojamiento de archivos, aunque también de publicación de sitios web estáticos. S3 también es una solución de respaldo. Glacier se convierte en una extensión de S3 para el archivado.
- **EBS**: *Elastic Block Store*: se trata de volúmenes de almacenado para los servidores EC2, o dicho de otra manera, los discos duros.

- **EFS**: *Elastic File System*: sistema de archivos NFS, y por lo tanto accesible simultáneamente por distintas instancias EC2.
- **VPC**: *Virtual Private Cloud*: es el servicio que permite crear una red privada y, por lo tanto, un cloud privado dentro de AWS. Se proporciona un rango de direcciones IP (CIDR) de /16 a /28, así como subnetting y tablas de enrutamiento. Los servidores y los servicios se vinculan a estas redes. Se puede hacer un peering hacia una red de empresa.
- **ELB**: *Elastic Load Balancer*: servicio de balanceo de carga que permite enrutar las conexiones hacia una o distintas instancias EC2, de manera pública (expuestas en Internet) o privada.
- **IAM**: *Identity and Access Management*: gestión de la seguridad de los accesos a los servicios AWS (cuentas, grupos, roles, gestión exhaustiva de los derechos, etc.).
- **KMS**: *Key Management Service*, para administrar las claves de cifrado.
- **SES**: *Simple Email Service*: servicio de tipo SMTP para los envíos de correos electrónicos.
- **SNS**: *Simple Notification Service*, servicio de notificación a través de mensajes SMS, por ejemplo, o envío de mensajes aplicativos (REST, cola de mensajes, etc.).
- **SQS**: *Simple Queue Service*, sistema de cola de mensajes, de tipo FIFO por ejemplo, un poco como Kafka, redis o rabbitMQ.
- **Route 53**: servicio de gestión de los servicios DNS.
- **Cloudfront**: un CDN (*Content Delivery Network*) para distribuir contenido en S3. Se trata de un sistema de caché que permite acelerar los accesos al contenido S3 en función de la ubicación geográfica.
- **Cloudwatch**: gestión y centralización de los registros y de las métricas de los distintos servicios (logs).
- **Cloudformation**: servicio de automatización y de orquestación de los recursos de AWS, bajo la forma de scripts.
- **RDS**: *Relational Database Service*: acceso a las bases de datos en modo SaaS, como MySQL, PostgreSQL, SQL Server, Oracle…
- **DynamoDB**: base de datos noSQL, compatible con MongoDB.
- **Lambda**: servicio de ejecución de código fuente sin servidor (serverless).

- **ECS**: *Elastic Container Service*, gestión de contenedores de tipo Docker.
- **Fargate**: un servicio de orquestación concebido para ECS.
- **EKS**: *Elastic Kubernetes Service*, un servicio de orquestación Kubernetes.

La lista de los servicios propuestos por los proveedores de soluciones Cloud está en constante evolución, según las necesidades de los clientes. Es uno de los componentes de su elasticidad.

El cloud, o la nube, es el conjunto de los servicios.

5. Zonas geográficas

AWS se extiende por distintas zonas geográficas, llamadas regiones, que se encuentran repartidas por el mundo. En Europa, por ejemplo:

- eu-central-1: Frankfurt
- eu-west-1: Dublín
- eu-west-2: Londres
- eu-west-3: París
- eu-north-1: Estocolmo

Algunos servicios están disponibles desde todas las regiones, otros no. De esta manera, un VPC de una determinada región no accederá al VPC de otra región, a no ser que haya mecanismos de peering entre las zonas. Así, un servidor EC2 situado en zona eu-central-1 no podrá comunicar con un servidor EC2 situado en eu-west-2. Sin embargo, un bucket S3 público podrá estar accesible desde cualquier sitio.

Cada región está dividida en zonas de disponibilidad (*availability zones*). Hay, como mínimo, dos por región. Estas zonas corresponden a la ubicación física de los servidores (los datacenters) para asegurar la disponibilidad de los servicios en caso de avería. He aquí las tres zonas para la región de Europa occidental:

- eu-west-1a: zona de disponibilidad a
- eu-west-1b: zona de disponibilidad b

- eu-west-1c: zona de disponibilidad c

Las zonas comunican entre ellas, pero aun así, por razones evidentes, algunos servicios son monozona, como por ejemplo los volúmenes EBS, que están directamente vinculados a las instancias EC2, y dependen de racks de discos locales. Pero podrá crear instantáneas (*snapshots*) que se podrán usar para crear un nuevo volumen EBS en otra zona.

Por lo tanto, lo más interesante es desplegar sus instancias EC2 en distintas zonas, por ejemplo, un repartidor ELB, para garantizar el acceso a sus servicios en caso de avería o mantenimiento.

La dirección postal de los centros de datos AWS no es pública, por razones de seguridad, pero cada zona corresponde a un centro de datos diferente.

6. Comprobar el estado

Bajo ciertas condiciones, puede comprobar fácil y gratuitamente los servicios cloud de Amazon. Ciertos tipos de instancias EC2 son gratuitas. Lo primero que tiene que hacer es crear una cuenta en https://aws.amazon.com/es/ y conectarse con su nombre de usuario y contraseña.

Descargue la CLI aquí: https://aws.amazon.com/es/cli/

Desde la consola web, usando IAM, empiece creando una cuenta que permitirá usar la API de AWS, dentro del grupo de administración. Obtendrá dos valores:

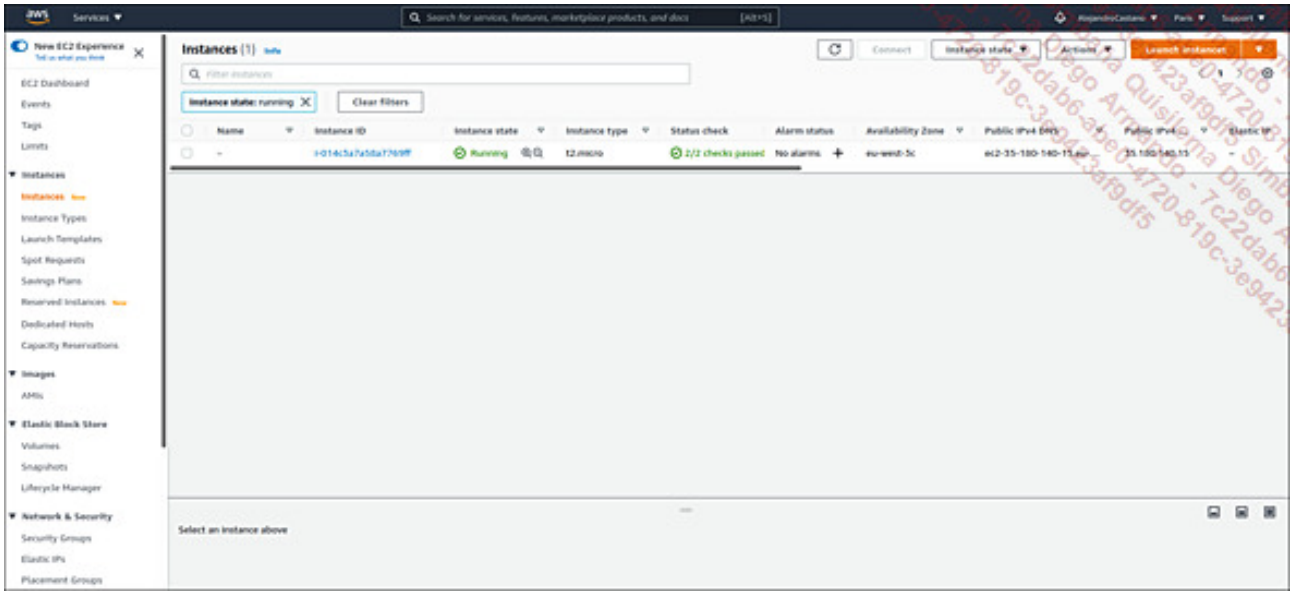
- La Key Id: nombre de usuario público de la cuenta.
- La Access Key: la clave privada de acceso, como una contraseña. No la comunique nunca a nadie.

Tiene que añadir su clave SSH pública para poder conectarse a su instancia una vez que esta esté ejecutándose.

Desde la consola web, cree una instancia de tipo t2.micro: es gratuita. Para ello, proceda de la manera siguiente:

- Elija una imagen **AMI**, la imagen de disco que contiene el sistema operativo que será arrancado en la instancia, una AMI dentro del plan gratuito como Ubuntu Server 18.04 LTS.
- Elija una instancia **t2.micro**, dentro del plan gratuito.
- Deje los valores por defecto en la configuración de la instancia.
- Deje los valores por defecto para el almacenamiento.
- Deje los valores por defecto para las etiquetas.
- Deje los valores por defecto para los grupos de seguridad: deberá acceder a su instancia usando SSH desde su puesto de trabajo.
- Añada su clave SSH para poder acceder a su instancia cuando se esté ejecutando.

Su instancia arrancará. Desde la consola EC2 verá algo parecido a la siguiente captura de pantalla:



Detalle de una instancia EC2

Verá que la instancia tiene una dirección IP pública, así como un nombre DNS público. Podrá acceder a ella desde su puesto de trabajo, usando SSH, conectándose a esta dirección:

```
$ ssh ubuntu@35.180.101.101
The authenticity of host '35.180.101.101 (35.180.101.101)' can't be established.
ECDSA key fingerprint is SHA256:KVlRURrML0in6AIWsA68k39RDvt1EHUoYZQvFc43cd4.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '35.180.101.101' (ECDSA) to the list of known hosts.
Enter passphrase for key '/Users/seb/.ssh/id_rsa':
Welcome to Ubuntu 18.04.3 LTS (GNU/Linux 4.15.0-1051-aws x86_64)
...

 System information as of Fri Jan  3 09:41:35 UTC 2020

 System load:  0.01            Processes:           88
 Usage of /:   13.6% of 7.69GB   Users logged in:     0
 Memory usage: 14%             IP address for eth0: 172.31.20.115
 Swap usage:   0%

0 packages can be updated.
0 updates are security updates.
...

ubuntu@ip-172-31-20-115:~$
```

¡Enhorabuena! ¡Acaba de crear su primer servidor en el Cloud de Amazon!

Ahora puede utilizarlo como cualquier servidor Linux en Ubuntu: instalar paquetes, crear cuentas, ejecutar servicios, etc. También puede controlar el acceso a su servidor actualizando su grupo de seguridad, añadir almacenamiento, cambiar el tipo de instancia, etc. Tenga precaución, ya que algunas opciones son de pago: es así como Amazon gana dinero.

➔ Configure ahora su CLI AWS en su puesto de trabajo:

```
$ aws configure
AWS Access Key ID [None]: AKIA5U3NHXFAY5GJIUDW
AWS Secret Access Key [None]: xxx/oM5xxx
Default region name [None]: eu-west-3
Default output format [None]:
```

Esto creará un repertorio .aws en la raíz de su cuenta con dos archivos, config y credentials. Compruebe el comando haciendo una lista solamente de las instancias de tipo t2.micro y mostrando su identificador:

```
$ aws ec2 describe-instances --filters "Name=instance-type,Values=t2.micro"
--query "Reservations[].Instances[].InstanceId"
[
  "i-03fcc45f4597fb2cd"
]
```

➔ Pare esta instancia:

```
$ aws ec2 stop-instances --instance-ids i-03fcc45f4597fb2cd
{
  "StoppingInstances": [
    {
      "CurrentState": {
        "Code": 64,
        "Name": "stopping"
      },
      "InstanceId": "i-03fcc45f4597fb2cd",
      "PreviousState": {
        "Code": 16,
        "Name": "running"
      }
    }
  ]
}
```

En la consola web, si actualiza la página, verá que el estado cambia de "stopping" a "stopped".

➔ Reiníciela:

```
$ aws ec2 start-instances --instance-ids i-03fcc45f4597fb2cd
{
```

```
    "StartingInstances": [
        {
            "CurrentState": {
                "Code": 0,
                "Name": "pending"
            },
            "InstanceId": "i-03fcc45f4597fb2cd",
            "PreviousState": {
                "Code": 80,
                "Name": "stopped"
            }
        }
    ]
}
```

En la consola, notará, después de reiniciar la instancia que la dirección IP pública ha cambiado. Esto es normal. Amazon asigna una dirección pública temporal para la duración del funcionamiento del servidor: un reinicio del sistema no afectará a la dirección IP. Cuando se detiene el servidor, la IP temporal se elimina y se asigna una nueva al reiniciar. Si lo desea, Amazon le puede vender direcciones IP públicas estáticas.

Con la CLI es posible programar, usando una API, la recuperación de esta dirección IP para actualizar automáticamente un servicio DNS local o el archivo de resolución local /etc/hosts…

➔ Pare y destruya la instancia, desde la CLI o desde la consola web:

```
$ aws ec2 terminate-instances --instance-ids i-03fcc45f4597fb2cd
{
    "TerminatingInstances": [
        {
            "CurrentState": {
                "Code": 32,
                "Name": "shutting-down"
            },
            "InstanceId": "i-03fcc45f4597fb2cd",
            "PreviousState": {
                "Code": 16,
```

```
            "Name": "running"
          }
        }
      ]
    }
```

7. cloud-init

Una imagen de instancia, sea cual sea el proveedor de solución Cloud, es un snapshot funcional de una imagen de un sistema operativo (que contiene una aplicación o no). En la creación de una instancia EC2, se crearán uno o más volúmenes EBS desde estas instantáneas. Como para cualquier máquina virtual, se pueden efectuar cierto número de acciones para modificar algunos valores al reinicio, como por ejemplo:

- el nombre de la máquina,
- la configuración de la red (dirección MAC, DHCP),
- las claves ssh de la máquina en .ssh/authorized_keys con la clave que envió al crearla,
- etc.

Es necesario que un servicio especial se ejecute, como uno de los primeros servicios del sistema, en el primer arranque de la instancia para implementar esta configuración.

El servicio cloud-init cubre esta necesidad. Se describe aquí: https://cloud-init.io/. Desarrollado por Canonical, creador de Ubuntu, se ha convertido en el estándar para todos los sistemas operativos de tipo Linux y para todos los proveedores de soluciones de cloud.

Cuando se crea una instancia, cloud-init modificará la configuración del sistema operativo, especialmente las tareas listadas arriba. El usuario podrá modificar su configuración para añadir, por ejemplo, paquetes o ejecutar comandos de shell.

Encontrará la documentación de cloud-init aquí: https://cloudinit.readthedocs.io/en/latest/

He aquí un ejemplo de configuración personalizado que instalará un paquete apache, lo

arrancará y creará una página estática:

```
#cloud-config
repo_update: true
repo_upgrade: all

packages:
- httpd

runcmd:
- systemctl start httpd
- sudo systemctl enable httpd
- [ sh, -c, "usermod -a -G apache ec2-user" ]
- [ sh, -c, "chown -R ec2-user:apache /var/www" ]
  - chmod 2775 /var/www
  - [ find, /var/www, -type, d, -exec, chmod, 2775, {}, \; ]
  - [ find, /var/www, -type, f, -exec, chmod, 0664, {}, \; ]
  - [ sh, -c, 'echo "Hello" > /var/www/html/index.html' ]
```

Durante la creación de la instancia AWS, en la página relativa a las opciones, podrá modificar los valores de los datos de los usuarios y añadir su propio script, en forma de archivo. Este se añadirá a las acciones por defecto definidas en esta AMI.

Puede modificar totalmente la configuración de cloud-init, pero deberá pasar por el CLI o la API para crear su instancia.