AMAZON WEB SERVICES

**SEC-3: ¿QUÉ ES EL CLOUD COMPUTING?**

* El Cloud Computing (Computación en la nube) es el suministro bajo demanda de potencia de cálculo, almacenamiento en bases de datos, aplicaciones y otros recursos informáticos.
* A través de una plataforma de servicios en el cloud con precios de pago por uso.
* Puedes aprovisionar exactamente el tipo y el tamaño de los recursos informáticos que necesitas.
* Puedes acceder a tantos recursos como necesites, casi al instante.
* Forma sencilla de acceder a servidores, almacenamiento, bases de datos y un conjunto de servicios de aplicaciones.
* Amazon Web Services (AWS) posee y mantiene el hardware conectado a la red necesario para estos servicios de aplicaciones, mientras que aprovisionas y utilizas lo que necesitas a través de una aplicación web.

**Los modelos de despliegue del Cloud:**

Cloud privado:

* Servicios en el cloud utilizados por una sola organización, no expuestos al público.
* Control total.
* Seguridad para aplicaciones sensibles.
* Satisfacer necesidades empresariales específicas.

Cloud público:

* Recursos en el cloud que son propiedad de un proveedor de servicios en el cloud y son operados por él, y que se suministran a través de Internet.
* Seis ventajas de la computación en el cloud.

Cloud híbrido:

* Mantener algunos servidores en las instalaciones y extiende algunas capacidades al cloud.
* Control de los activos sensibles en tu infraestructura privada.
* Flexibilidad y rentabilidad del cloud público.

**Las cinco características del cloud computing:**

Autoservicio bajo demanda:

* Los usuarios pueden aprovisionar recursos y utilizarlos sin interacción humana del proveedor de servicios.

Amplio acceso a la red:

* Los recursos están disponibles a través de la red, y pueden ser accedidos por diversas plataformas de clientes.

Alquiler múltiple y agrupación de recursos:

* Varios clientes pueden compartir la misma infraestructura y aplicaciones con seguridad y privacidad.
* Múltiples clientes reciben servicio desde los mismos recursos físicos.

Rápida elasticidad y escalabilidad:

* Adquirir y disponer de recursos de forma automática y rápida cuando sea necesario.
* Escala rápida y fácilmente en función de la demanda.

Servicio medido:

* El uso se mide, los usuarios pagan correctamente por lo que han utilizado.

**DIFERENTES TIPOS DE CLOUD COMPUTING**

**Infraestructura como servicio (IaaS):**

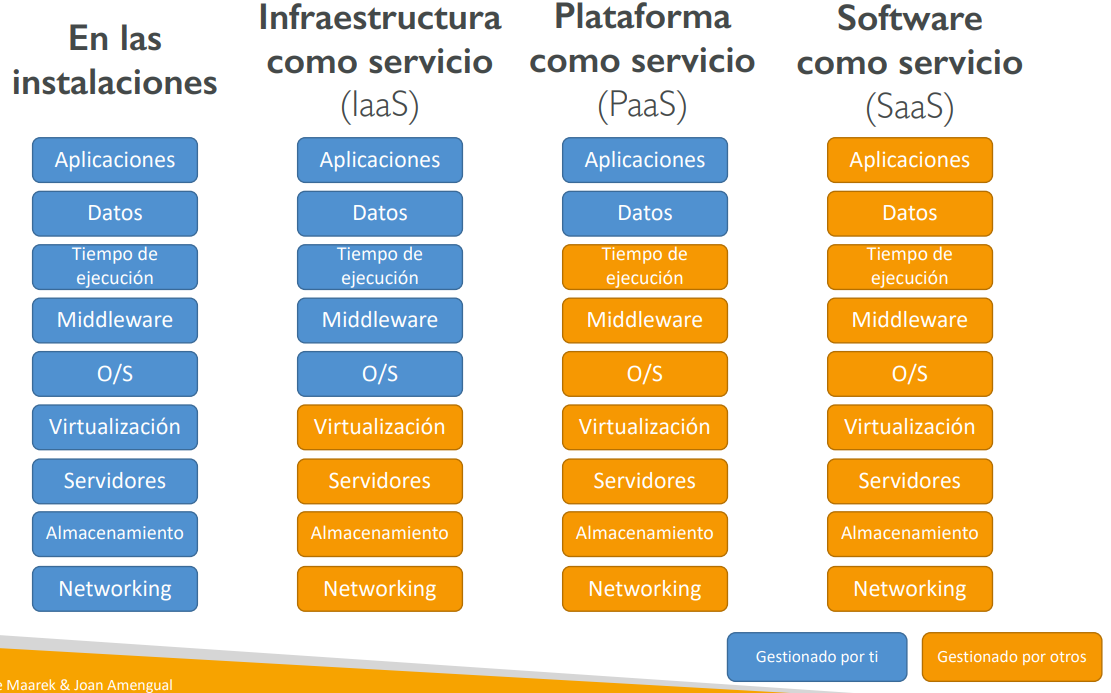
* Proporciona bloques de construcción para la IT en el cloud.
* Proporciona redes, ordenadores y espacio de almacenamiento de datos.
* Máximo nivel de flexibilidad.
* Fácil paralelismo con la IT tradicional en las instalaciones.

**Plataforma como servicio (PaaS):**

* Elimina la necesidad de que tu organización gestione la infraestructura subyacente.
* Se centra en el despliegue y la gestión de tus aplicaciones.

**Software como servicio (SaaS):**

* Producto completo que es ejecutado y gestionado por el proveedor de servicios.



**EJEMPLOS:**

**Infraestructura como servicio:**

* Amazon EC2 (en AWS).
* GCP, Azure, Rackspace, Digital Ocean, Linode.

**Plataforma como servicio:**

* Elastic Beanstalk (en AWS).
* Heroku, Google App Engine (GCP), Windows Azure (Microsoft).

**Software como servicio:**

* Muchos servicios de AWS (por ejemplo, Rekognition para el aprendizaje automático).
* Google Apps (Gmail), Dropbox, Zoom.

**VISTA GENERAL DEL CLOUD EN AWS**

* AWS permite crear aplicaciones sofisticadas y escalables.
* Aplicable a un conjunto diverso de industrias

Los casos de uso incluyen:

* Enterprise IT, copias de seguridad y almacenamiento, análisis de Big Data.
* Alojamiento de sitios web, aplicaciones móviles y sociales.
* Juegos.

**INFRAESTRUCTURA GLOBAL DE AWS**

**Regiones:**

* AWS tiene regiones en todo el mundo.
* Una región es un grupo de centro de datos.
* La mayoría de los servicios de AWS son de ámbito regional.

**¿Cómo elegir una región de AWS?**

* Cumplimiento de los requisitos legales y de gobernanza de datos: los datos nunca salen de una región sin tu permiso explícito.
* Proximidad a los clientes: latencia reducida.
* Servicios disponibles en una región: los nuevos servicios y las nuevas funciones no están disponibles en todas las regiones.
* Precios: los precios varían de una región a otra y son transparentes en la página de precios del servicio.

**Zona de disponibilidad de AWS:**

* Cada región tiene muchas zonas de disponibilidad (normalmente 3, el mínimo es 2, el máximo es 6). Ejemplo: ap-sudeste-2ª, ap-sudeste-2b, ap-sudeste-2c.
* Cada zona de disponibilidad (AZ) es uno o varios centros de datos discretos con alimentación, red y conectividad redundantes.
* Están separadas unas de otras, de modo que están aisladas de las catástrofes.
* Están conectadas con redes de alto ancho de banda y latencia ultra baja.

**Puntos de presencia de AWS:**

* Amazon tiene 216 puntos de presencia (205 puntos de presencia y 11 cachés regionales) en 84 ciudades de 42 países.
* El contenido se entrega a los usuarios finales con menor latencia.

**Tour por la consola de AWS:**

AWS cuenta con servicios globales:

* Identity and Access Management (IAM).
* Route 53 (servicio DNS).
* CloudFront (Red de entrega de contenido).
* WAF (Firewall de aplicaciones web)

La mayoría de los servicios de AWS son de ámbito regional:

* Amazon EC2 (Infraestructura como servicio).
* Elastic Beanstalk (Plataforma como servicio).
* Lambda (Función como servicio).
* Rekognition (Software como servicio).

Tabla de regiones:

<https://aws.amazon.com/about-aws/global-infrastructure/regional-product-services>

**MODELO DE RESPONSABILIDAD COMPARTIDA Y POLÍTICA ACEPTABLE DE AWS**



**Política de uso aceptable de AWS:**

<https://aws.amazon.com/es/aup/>

* Ningún uso o contenido ilegal, dañino u ofensivo.
* No a las violaciones de la seguridad.
* No al abuso de la red.
* No al abuso de correo electrónico u otros mensajes.

**SEC-4: IAM – IDENTITY and ACCESS MANAGEMENT**

**INTRODUCCIÓN A IAM: Usuarios, Grupos y Políticas**

* IAM = Identity and Access Management, servicio global.
* Cuenta root / raíz creada por defecto, no debe ser utilizada ni compartida.
* Los usuarios son personas dentro de tu organización, y pueden ser agrupados.
* Los grupos sólo contienen usuarios, no otros grupos.
* Los usuarios no tienen que pertenecer a un grupo, y el usuario puede pertenecer a varios grupos.

**Permisos:**

* A los usuarios o grupos se les pueden asignar documentos JSON llamados políticas.
* Estas políticas definen los permisos de los usuarios.
* En AWS se aplica el principio de mínimo privilegio: no dar más permisos de los que un usuario necesita.



**USUARIOS Y GRUPOS DE AIM**

No se recomienda usar nuestra cuenta raíz, por lo que recomienda crear un usuario para el servicio de IAM.

Para ello ya ubicados en el servicio IAM en AWS, vamos al menú lateral izquierdo a la opción **usuarios**. Y luego seleccionamos **agregar usuario**.

Ahora debemos darle un nombre de usuario y una contraseña.

A continuación, creamos un grupo dándole un nombre y elegimos que permisos puede tener.

Luego se le puede añadir una etiqueta. Por ejemplo, pertenece un departamento en particular, ingenieros, etc.

Y listo, ya el usuario estaría creado. No olvidarse de ver la contraseña si la creo automáticamente AWS.

**ACCESO COMO USUARIO IAM**

Seleccionamos en el menú lateral izquierdo la opción de **Panel**. Ahora vamos a ver que en la parte de la derecha hay un título **Cuenta de AWS**, clickeamos donde dice **alias de cuenta** en **crear**.

Ahora ya creado este **alias**, un poco más abajo hay un link donde aparece el alias. Copiemos ese link.

Este link nos va a permitir entrar como usuario de IAM.

**POLÍTICAS IAM**

**Estructura de las políticas IAM:**

Consta de:

* Version: versión del lenguaje de la política, siempre incluye "2012-10-17".
* Id: un identificador para la política (opcional).
* Statement: una o más declaraciones individuales (obligatorio).

Las declaraciones constan de:

* Sid: un identificador para la declaración (opcional).
* Effect: si la sentencia permite o deniega el acceso (Permitir, Denegar).
* Principal: cuenta/usuario/rol al que se aplica esta política.
* Action: lista de acciones que esta política permite o deniega.
* Resource: lista de recursos a los que se aplican las acciones.
* Condition: condiciones para cuando esta política está en efecto (opcional).



**POLÍTICAS AIM- PRÁCTICA**

Podemos agregar a un usuario a múltiples grupos, y además podemos crear políticas yendo al menú lateral izquierdo en **Políticas**. Allí podemos elegir el servicio, las acciones, etc.

**VISIÓN GENERAL DE IAM MFA**

**Política de contraseñas:**

Contraseñas fuertes = mayor seguridad para tu cuenta.

En AWS, puedes configurar una política de contraseñas:

* Longitud mínima de contraseña.
* Requerir tipos de caracteres específicos (letras mayúsculas, letras minúsculas, números, caracteres no alfanuméricos).
* Permitir a todos los usuarios de IAM cambiar sus propias contraseñas.
* Requerir a los usuarios que cambien su contraseña después de un tiempo (caducidad de la contraseña).
* Impedir la reutilización de la contraseña.

**Multi Factor Authentication – MFA:**

* Los usuarios tienen acceso a tu cuenta y posiblemente pueden cambiar configuraciones o eliminar recursos en tu cuenta de AWS.
* Quieres proteger tus cuentas root y los usuarios de IAM.
* MFA = contraseña que conoces + dispositivo de seguridad que posees.
* Principal beneficio de MFA: si una contraseña es robada o hackeada, la cuenta no se ve comprometida.

**Opciones de dispositivo:**

* Dispositivo virtual de MFA: Autenticador de Google, Authy.
* Clave de seguridad del segundo factor universal (U2F): Yubikey.
* Dispositivo MFA de llavero por hardware.
* Dispositivo MFA de llavero por hardware para AWS GovCloud (US).

**IAM MFA – PRÁCTICA**

Dentro del servicio IAM ir a **configuracion de cuenta** en el menú lateral izquierdo. Luego dentro de esta sección vamos a poder editar la política de las contraseñas, especificando como deben ser las mismas.

Ahora en la esquina superior derecha donde esta nuestro nombre clickear en **credenciales de seguridad**. Allí clickeamos en **asignar MFA**, nos dará para elegir por cual método, y elegimos por Google authenticator por ejemplo. Luego seguimos los pasos y activamos el MFA.

Ya con esto activado, cuando queramos volver a ingresar a AWS vamos a necesitar el token de autenticación.

**CLAVES DE ACCESO DE AWS, CLI Y SDK**

Para acceder a AWS tienes 3 opciones:

* Consola de administración de AWS (protegida por contraseña + MFA).
* Interfaz de línea de comandos de AWS (CLI): protegida por claves de acceso.
* AWS Software Developer Kit (SDK) - para el código: protegido por claves de acceso.

Las claves de acceso se generan a través de la consola de AWS.

Los usuarios gestionan sus propias claves de acceso.

Las claves de acceso son secretas, como una contraseña. No las compartas.

ID de la clave de acceso ~= nombre de usuario.

Clave de acceso secreta ~= contraseña.

**CLI de AWS:**

* Una herramienta que permite interactuar con los servicios de AWS mediante comandos en tu shell de línea de comandos.
* Acceso directo a las API públicas de los servicios de AWS.
* Puedes desarrollar scripts para gestionar tus recursos.
* Es de código abierto <https://github.com/aws/aws-cli>
* Alternativa al uso de la consola de administración de AWS.

**SDK de AWS:**

* Kit de desarrollo de software de AWS (AWS SDK).
* APIs específicas para cada lenguaje (conjunto de bibliotecas).
* Permite acceder y administrar los servicios de AWS mediante programación.
* Integrado en la aplicación.
* Admite: SDKs (JavaScript, Python, PHP, .NET, Ruby, Java, Go, Node.js, C++) • SDKs para móviles (Android, iOS, ...), SDKs para dispositivos IoT (Embedded C, Arduino, ...).
* Ejemplo: AWS CLI está construido sobre AWS SDK para Python.

**CONFIGURACIÓN de la CLI de AWS en WINDOWS**

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

Los pasos a seguir están en la documentación, es solo descargar lo que dice ahí y verificarlo en la terminal con el comando **aws --version**.

**CONFIGURACIÓN de la CLI de AWS en MAC**

Seguir de la documentación de arriba el **GUI Installer**, el cual nos hace descargar un archivo **pkg**, le damos a todo continuar y después solamente en la terminal ver la ruta donde lo instalo con el comando **which aws** y la versión con el comando **aws --versión**.

**CLI DE AWS – PRÁCTICA**

* Dentro del servicio **IAM** nos dirigimos a **usuarios** en el menú lateral izquierdo.
* Elegimos el usuario y vamos donde dice **credenciales de seguridad**.
* Dentro de esta sección nos vamos al título **Claves de acceso**, clickeamos donde dice **crear una clave de acceso**.
* Una vez la tengamos abrimos la terminal y escribimos el comando **aws configure** para configurar la clave de acceso. Este comando nos pedirá el **id** y luego el **secret** (el default región name y el default output format lo podemos dejar como están).
* Ahora ya podemos usar los comandos cli.
* Probar el comando **aws iam list-users**, el cual nos devolverá todos los usuarios iam que existan.

**AWS CLOUDSHELL**

Este servicio funciona solamente para ciertas regiones y nos permite acceder a la terminal de una forma más genérica.

Es una terminal como cualquiera, donde podemos crear archivos, usar el comando **ls** para ver que tenemos en la carpeta, etc.

Arriba a la derecha hay un **select** con la palabra **actions**, aquí podemos descargar archivos, actualizarlos, entre otras funciones.

Para descargar un archivo debemos escribir el **path** donde se encuentra el archivo (lo podemos ver con el comando **pwd**) + /nombre del archivo con su extensión. Y clickeamos en descargar y listo.

**ROLES DE IAM PARA LOS SERVICIOS DE AWS**

Se le puede otorgar a un usuario un permiso para que utilice cierto servicio de AWS.

Dentro del servicio **IAM** en el menú lateral izquierdo nos vamos a la sección **Roles**.

Una vez aquí adentro vamos a **Crear Rol**.

Donde dice **Entidad de confianza** elegimos **servicio de AWS**, luego en **Caso de Uso**, elegimos el servicio que vamos a permitir. Por ejemplo, en este caso **EC2**.

Damos click en **siguiente**.

Ahora elegimos la **política de permisos**. En este ejemplo usamos **IAMReadOnlyAccess**. Y damos click en siguiente.

En esta sección le damos un nombre al rol y lo creamos.

**HERRAMIENTAS DE SEGURIDAD DE IAM**

Hay dos herramientas importantes:

* **IAM Credentials Report / Informe de credenciales de IAM** (a nivel de cuenta): nos da un informe que enumera todos los usuarios de tu cuenta y el estado de tus diversas credenciales.
* **IAM Access Advisor / Asesor de acceso de IAM** (a nivel de usuario): muestra los permisos de servicio concedidos a un usuario y cuando se accedió a esos servicios por última vez. Puedes utilizar esta información para revisar tus políticas concedidas.

**HERRAMIENTAS DE SEGURIDAD DE IAM – PRÁCTICA**

Dentro del servicio **IAM** en el menú lateral izquierdo vamos a la sección **Informe de credenciales**, en el cual podemos descargar el informe que nos dará la información de todos los usuarios que tengamos.

Para ver el **Access Advisor** vamos a **usuarios**, elegimos el usuario, y vamos a la sección **Access Advisor**, y allí veremos los servicios que uso y cuando se accedió.

**MEJORES PRÁCTICAS DE IAM**

* No utilices la cuenta root excepto para la configuración de la cuenta AWS.
* Un usuario físico = Un usuario AWS.
* Asignar usuarios a grupos y asignar permisos a grupos.
* Crear una política de contraseñas fuerte.
* Utilizar y reforzar el uso de la autenticación multifactor (MFA).
* Crear y utilizar Roles para dar permisos a los servicios de AWS.
* Utilizar claves de acceso para el acceso programático (CLI / SDK).
* Revisar los permisos de tu cuenta con el informe de credenciales de IAM.
* No compartir nunca los usuarios de IAM ni las claves de acceso.

**MODELO DE RESPONSABILIDAD COMPARTIDA PARA IAM**

Responsabilidad de AWS:

* Infraestructura (seguridad de la red global).
* Análisis de configuración y vulnerabilidad.
* Validación de la conformidad.

Responsabilidad nuestra:

* Gestión y supervisión de usuarios, grupos, roles y políticas.
* Habilitar MFA en todas las cuentas.
* Rota todas tus claves con frecuencia.
* Utiliza las herramientas IAM para aplicar los permisos adecuados.
* Analiza los patrones de acceso y revisa los permisos.

**RESUMEN**

* **Usuarios:** mapeado a un usuario físico, tiene una contraseña para la consola de AWS.
* **Grupos:** contiene sólo usuarios.
* **Políticas:** Documento JSON que describe los permisos para los usuarios o grupos.
* **Roles:** para instancias EC2 o servicios AWS.
* **Seguridad:** MFA + Política de contraseñas.
* **AWS CLI:** gestiona tus servicios de AWS mediante la línea de comandos.
* **AWS SDK:** gestiona tus servicios de AWS utilizando un lenguaje de programación.
* **Claves de acceso:** accede a AWS mediante la CLI o el SDK.
* **Auditoría:** Informes de credenciales de IAM y Asesor de acceso de IAM.

**SEC-5: EC2 – ELASTIC COMPUTE CLOUD**

**CONFIGURACIÓN DEL PRESUPUESTO DE AWS**

Para ello, vamos a ingresar a AWS con el usuario **raíz**, dentro del menú desplegable ubicado en nuestro nombre de usuario, nos dirigimos a **cuenta**. Una vez allí bajamos hasta encontrar el título **Acceso de los usuarios y los roles de IAM a la información de facturación**.

Seleccionamos **editar** y habilitamos el checkbox **Active el acceso de IAM**, y le damos **actualizar**.

Ahora volvemos a ingresar a AWS con el usuario de IAM (recordar que es buena práctica usar siempre el usuario IAM).

Nos dirigimos en el menú desplegable de nuestro nombre al **Panel de facturación**. Dentro de esta en el menú lateral izquierdo podríamos ir a **Facturación** y allí encontrar todos los gastos que hemos hechos en AWS.

**Budgets:**

Nos dirigimos a esta sección para poder crear un presupuesto de lo que vamos a gastar en servicios, y así no correr el riesgo de activar un servicio demasiado caro.

Clickeamos en **Crear un presupuesto** y elegimos el plan que queremos, cuanto queremos gastar, el nombre del presupuesto, etc.

También podemos agregar umbrales de alarma, para saber cuando estamos llegando al presupuesto elegido, por ejemplo.

**FUNDAMENTOS DE EC2**

Consiste principalmente en la capacidad de:

* Alquilar máquinas virtuales (EC2).
* Almacenar datos en unidades virtuales (EBS).
* Distribuir la carga entre las máquinas (ELB).
* Escalar los servicios mediante un Auto Scaling Group (ASG) o también conocido en español como Grupo de Autoescalado.

**Opciones de tamaño y configuración de EC2:**

* Sistema operativo (OS): Linux, Windows o Mac OS.
* Cuánta potencia de cálculo y núcleos (CPU).
* Cuánta memoria de acceso aleatorio (RAM).
* Cuánto espacio de almacenamiento: Conectado a la red (EBS y EFS) o hardware (EC2 Instance Store).
* Tarjeta de red: velocidad de la tarjeta, dirección IP pública.
* Reglas de firewall: grupo de seguridad.
* Script de arranque (configurar en el primer lanzamiento): Datos de usuario de EC2

**Datos del usuario de EC2:**

* Es posible arrancar nuestras instancias utilizando un script de datos de usuario de EC2.
* Bootstrapping significa lanzar comandos cuando una máquina se inicia.
* Ese script sólo se ejecuta una vez en el primer arranque de la instancia.
* Los datos de usuario de EC2 se utilizan para automatizar tareas de arranque como: Instalar actualizaciones, Instalación de software, Descarga de archivos comunes de Internet, Cualquier cosa que se te ocurra.
* El script de datos de usuario de EC2 se ejecuta con el usuario root.

**CREAR UNA INSTANCIA EC2 CON DATOS DE USUARIO PARA TENER UN SITIO WEB**

Dentro del servicio **EC2** vamos ir a **instancias** en el menú lateral izquierdo. Una vez aquí adentro clickeamos en **lanzar instancia**.

* Elegimos el **nombre** de la instancia.
* Luego el inicio rápido con **amazon Linux**.
* Ahora debemos crear un **par de claves** donde le damos un nombre, elegimos el tipo **RSA** y por último el formato de archivo de clave privada, que va a depender de si usamos Linux, mac o windows10 será **.pem**, en caso de que utilicemos una versión menor a windows10 utilizaremos **.pkk**.
* Ahora en **Configuración de la red** debemos activar el checkbox de **permitir el tráfico de HTTP desde Internet**.
* En **Detalles avanzados** vamos a donde dice **datos de usuario** y pegamos las siguientes líneas de código:

#!/bin/bash

# Utiliza esto para tus datos de usuario

# Instala httpd (Version: Linux 2)

yum update -y

yum install -y httpd

systemctl start httpd

systemctl enable httpd

echo "<h1>Hola Mundo desde $(hostname -f)</h1>" > /var/www/html/index.html

* Y ahora clickeamos en **Lanzar instancia**.

Listo ya tenemos nuestra primera instancia lanzada.

Ahora vamos nuevamente a la pestaña **instancias** y seleccionamos la instancia creada. Esto nos va a abrir un panel por debajo con toda la información de la misma.

En la sección **Seguridad** en el título **Reglas de entrada** debemos tener en **intervalos de puertos** los números 22 y 80 del **protocolo** TCP con el **grupo de seguridad** launch-wizard-1 en este caso.

En la sección **Detalles** podemos copiar la **dirección IPv4 pública** y acceder a nuestro mensaje que escribimos en **datos de usuario**.

Además, podemos detener, reiniciar, iniciar y terminar la instancia desde el **select** que esta arriba de todo que dice **Estado de la instancia**.

**TIPOS BÁSICOS DE INSTANCIAS EC2**

Puedes utilizar diferentes tipos de instancias EC2 optimizadas para diferentes casos de uso <https://aws.amazon.com/ec2/instance-types/>

AWS tiene la siguiente convención de nombres: m5.2xlarge

* **m**: clase de instancia.
* **5**: generación (AWS los mejora con el tiempo).
* **2xlarge**: tamaño dentro de la clase de instancia.

**Propósito general:** (T2)

* Excelente para una diversidad de cargas de trabajo, como servidores web o repositorios de código.
* Equilibrio entre: Computación, Memoria y Red.
* En el curso, utilizaremos la instancia t2.micro que es una instancia EC2 de propósito general.

**Computación optimizada:** (C6g)

Ideal para tareas de cálculo intensivo que requieren procesadores de alto rendimiento:

* Cargas de trabajo de procesamiento por lotes.
* Transcodificación de medios.
* Servidores web de alto rendimiento.
* Computación de alto rendimiento (HPC).
* Modelado científico y aprendizaje automático.
* Servidores dedicados a juegos.

**Memoria optimizada:** (R6g).

Rápido rendimiento para cargas de trabajo que procesan grandes conjuntos de datos en memoria.

Casos de uso:

* Alto rendimiento, bases de datos relacionales/no relacionales.
* Almacenes de caché distribuidos a escala web.
* Bases de datos en memoria optimizadas para BI (business intelligence).
* Aplicaciones que realizan el procesamiento en tiempo real de grandes datos no estructurados.

**Almacenamiento optimizado:** (I3)

Ideal para tareas de almacenamiento intensivo que requieran un acceso alto y secuencial de lectura y escritura a grandes conjuntos de datos en el almacenamiento local.

Casos de uso:

* Sistemas de procesamiento de transacciones en línea (OLTP) de alta frecuencia.
* Bases de datos relacionales y NoSQL.
* Caché para bases de datos en memoria (por ejemplo, Redis).
* Aplicaciones de almacenamiento de datos.
* Sistemas de archivos distribuidos.

**GRUPOS DE SEGURIDAD Y PUERTOS CLÁSICOS**

* Los grupos de seguridad son la base de la seguridad de la red en AWS.
* Controlan cómo se permite el tráfico dentro o fuera de nuestras Instancias EC2.
* Los grupos de seguridad sólo contienen reglas de permiso.
* Las reglas de los grupos de seguridad pueden hacer referencia por IP o por grupo de seguridad.
* Los grupos de seguridad actúan como un “firewall" en las instancias de EC2.

Regulan:

* El acceso a los puertos.
* Rangos de IP autorizados - IPv4 e IPv6.
* Control de la red de entrada (de otros a la instancia).
* Control de la red saliente (desde la instancia hacia otra).

**Es bueno saber:**

* Puede adjuntarse a múltiples instancias.
* Bloqueado a una combinación de región / VPC.
* Vive "fuera" del EC2 - si el tráfico está bloqueado, la instancia EC2 no lo verá.
* Es bueno mantener un grupo de seguridad separado para el acceso SSH.
* Si tu aplicación no es accesible (tiempo de espera), entonces es un problema de grupo de seguridad.
* Si tu aplicación da un error de "conexión rechazada", entonces es un error de la aplicación o no se ha lanzado.
* Todo el tráfico de entrada está bloqueado por defecto.
* Todo el tráfico de salida está autorizado por defecto.

**Puertos clásicos que hay que conocer:**

* 22 = SSH (Secure Shell) - iniciar sesión en una instancia de Linux.
* 21 = FTP (File Transfer Protocol) - subir archivos a un archivo compartido.
* 22 = SFTP (Secure File Transfer Protocol) - subir archivos usando SSH.
* 80 = HTTP - acceso a sitios web no seguros.
* 443 = HTTPS - acceso a sitios web seguros.
* 3389 = RDP (Remote Desktop Protocol) - iniciar sesión en una instancia de Windows.

**GRUPOS DE SEGURIDAD – PRÁCTICA**

Creamos una instancia, le damos el nombre, elegimos el par de claves ya creado con anterioridad, en **configuración de red** elegimos un grupo de seguridad existente, y ahora nos vamos a **detalles avanzados** a datos de usuarios (son los comandos que se van a activar una vez que se levante la instancia) y copiamos el siguiente código:

#!/bin/bash

# Utiliza esto para tus datos de usuario

# Instala httpd (Version: Linux 2)

yum update -y

yum install -y httpd

systemctl start httpd

systemctl enable httpd

echo "<h1>Hola Mundo desde $(hostname -f)</h1>" > /var/www/html/index.html

Y ahora lanzamos la instancia.

A continuación, en el título **Red y seguridad** del menú lateral izquierdo seleccionamos la sección **Security Groups**.

Ahora nos mostrará todos los grupos de seguridad que tengamos, seleccionamos el **launch-wizard-1** que es el que elegimos cuando creamos la instancia, y se nos abrirá un panel con 4 secciones: detalles, reglas de entrada, reglas de salida y etiquetas.

Si seleccionamos **reglas de entrada** y nos vamos a **editar reglas de entrada**, veremos que en la parte de **origen** esta seleccionada **personalizada** con el valor en 0.0.0.0/0, lo cual significa que identifica a cualquier persona. Si borramos el protocolo HTTP ahora no podremos acceder a la instancia.

En **reglas de salida>editar reglas de salida** por defecto el tipo es **Todo el tráfico** hacia cualquier destino.

**VISIÓN GENERAL DE SSH**

El protocolo SSH nos va a servir para conectarnos directamente a las instancias de AWS a las F2.

Podemos usar SSH con Linux, Mac y Windows >= 10.

**CÓMO UTILIZAR SSH USANDO LINUX O MAC**

SSH es una de las funciones más importantes. Permite controlar una máquina remota, todo ello utilizando la línea de comandos.

Vamos al servicio **EC2** en el menú lateral izquierdo en la sección **Red y seguridad** clickeamos en **Pares de claves**, y ahora en **crear par de claves**.

Le damos el **nombre**, el **tipo** que sea **RSA** y el formato **.pem**.

Ahora con este par de claves creada vamos a crear una **instancia**.

Una vez creada la instancia, vamos a abrir una terminal y ejecutar el comando **ssh -i EC2Tutorial.pem**(nombre del par de claves creada) **ec2-user**(nombre)**@54.162.162.128**(dirección IPv4 pública).

Tenemos que estar en el directorio de la ubicación del par de claves con la terminal para ejecutar el comando.

Además, vamos a tener que cambiar unos permisos con el comando **chmod 04000 EC2Tutorial.pem** (nombre del par de claves creada).

Ahora deberíamos estar conectados, podemos con los comandos **whoami** saber que usuario soy, **exit** para salir de la instancia

**CÓMO UTILIZAR SSH USANDO WINDOWS <10**

SSH es una de las funciones más importantes. Permite controlar una máquina remota, todo ello utilizando la línea de comandos.

Necesitamos descargar **putty** en la versión 64.

Una vez descargado abrimos el menú de Windows y buscamos **PuTTYgen**, el cual nos va a permitir asignar la clave que tenemos para nuestra instancia.

Clickeamos en **File>Load private key** y buscamos el par de claves que debimos haber creado en el servicio de **EC2**.

Seleccionamos ahora **Save private key**, ahora nos convertirá el formato a (**.ppk**). Debemos darle un nombre, como, por ejemplo: **EC2Tutorial.ppk**.

Ahora nuevamente abrimos el menú de Windows y buscamos **PuTTY**, el cual nos abrirá un panel donde deberemos escribir el **IP** de la instancia creada en **EC2**, con el nombre de usuario adelante separando el IP por un **@**, ejemplo: [ec2-user@35.180.100.144](mailto:ec2-user@35.180.100.144). Luego guardamos la sesión dándole un nombre, como, por ejemplo: MyEC2Instance.

Seguramente ahora nos de un **error**, por lo que ahora debemos volver a abrir **PuTTY**, cargamos la sesión que acabamos de crear, clickeamos en **SSH>Auth** en el menú lateral del panel, la cual nos pedirá una clave privada para la autenticación.

Y ahora con esto si podremos conectarnos.

**CÓMO UTILIZAR SSH USANDO WINDOWS >10**

SSH es una de las funciones más importantes. Permite controlar una máquina remota, todo ello utilizando la línea de comandos.

Abrimos la **PowerShell** y ejecutamos el comando **ssh -i (ruta del par de claves) (ec2-user@IP de la instancia)**

Si ejecutamos esto nos va a dar un **error**, porque no tenemos los permisos. Lo que hacemos es ir al archivo del par de claves, **click derecho>propiedades>security**, clickeamos en **advanced** y donde dice **owner/propietario** clickeamos en cambiar y ponemos **Joaquin**, eliminamos el anterior y deshabilitamos la jerarquía de permisos.

Y ahora si debería andar el comando.

ssh -i “C:\Users\JOAQUÍN\Dropbox\Mi-PC-(DESKTOP-TD1DLMC)\Desktop\DEVELOPER\Cuentas\AWS\EC2Tutorial.pem” ec2-user@34.229.70.148

**EC2 INSTANCE CONNECT**

Nos permite acceder vía el navegador a una conexión SSH, independientemente si usas Windows, Linux o Mac.

Para ello vamos al servicio de **EC2**, seleccionamos la instancia y clickeamos en **Conectar**, seleccionamos **Conexión de la instancia EC2**, el nombre de usuario que sea **ec2-user** y **conectamos**.

**DEMOSTRACIÓN DE LOS ROLES DE LAS INSTANCIAS DE EC2**

Vamos al servicio **IAM** y en la sección **Administración del acceso**, clickeamos en **Roles**.

Ahora debemos ver que tengamos el **DemoRolEC2** que lo creamos en la sección anterior del curso y solo tiene un permiso de **IAMReadOnlyAccess**.

Con esto creado, volvemos a las instancias en **EC2** y vamos a configurar un rol para la instancia.

Nos conectamos a la instancia con **instance connect** y una vez dentro ya podemos introducir los comandos.

No vamos a poder usar el comando **aws** pero:

NO DEBEMOS NUNCA INTRODUCIR LAS CREDENCIALES DESDE INSTANCE CONNECT.

Ahora vamos a elegir la instancia y clickeamos en **Acciones>Seguridad>Modificar rol de IAM**.

Ahora seleccionamos **DemoRolEC2** y clickeamos en **Actualizar rol de IAM**.

Y nos conectamos nuevamente con la instancia. Ahora si vamos a poder usar el comando **aws**.

**OPCIONES DE COMPRA DE INSTANCIAS EC2**

* **Instancias bajo demanda:** carga de trabajo corta, precio predecible, pago por segundos.
* **Reservadas** (1 y 3 años): **Instancias reservadas** - cargas de trabajo largas, **Instancias reservadas convertibles** - cargas de trabajo largas con instancias flexibles.
* **Planes de ahorro** (1 y 3 años) - compromiso con una cantidad de uso, carga de trabajo larga.
* **Instancias Spot** - cargas de trabajo cortas, baratas, pueden perder instancias (menos fiables).
* **Hosts dedicados:** reserve un servidor físico completo, controle la ubicación de las instancias.
* **Instancias dedicadas:** ningún otro cliente compartirá tu hardware.
* **Reservas de capacidad:** reserva de capacidad en una AZ específica para cualquier duración.

**EC2 Bajo demanda:**

* Paga por lo que usas:

• Linux o Windows - facturación por segundo, después del primer minuto.

• Todos los demás sistemas operativos: facturación por hora.

* Tiene el coste más elevado, pero no hay que pagar por adelantado.
* Sin compromiso a largo plazo.
* Recomendado para cargas de trabajo a corto plazo y sin interrupciones, cuando no se puede predecir el comportamiento de la aplicación.

**Instancias reservadas de EC2:**

* Hasta un 72% de descuento en comparación con el servicio bajo demanda.
* Reserva de atributos de instancia específicos (tipo de instancia, región, ocupación, sistema operativo).
* Periodo de reserva - 1 año (+descuento) o 3 años (+++descuento).
* Opciones de pago - Sin pago inicial (+), Pago inicial parcial (++), Pago inicial total (+++).
* Alcance de la instancia reservada - Por región o por zona (capacidad de reserva en una AZ).
* Recomendado para aplicaciones de uso constante (piensa en una base de datos).
* Puedes comprar y vender en el Marketplace de instancias reservadas.
* Instancia reservada convertible:

• Puedes cambiar el tipo de instancia EC2, la familia de instancias, el SO, etc.

• Hasta un 66% de descuento.

**Planes de ahorro EC2:**

* Obtén un descuento basado en el uso a largo plazo (hasta el 72%).
* Comprométete a un determinado tipo de uso (10 $/hora durante 1 o 3 años).
* El uso más allá de los planes de ahorro de EC2 se factura al precio bajo demanda.
* Bloqueado a una familia de instancias específica y a una región de AWS (por ejemplo, M5 en us-east-1).
* Flexible a través de:

• Tamaño de instancia (por ejemplo, m5.xlarge, m5.2xlarge).

• Sistema operativo (por ejemplo, Linux, Windows).

• Tenencia (Anfitrión, Dedicado, Por defecto).

**Instancias EC2 Spot:**

* Puedes obtener un descuento de hasta el 90% en comparación con la demanda.
* Instancias que puedes "perder" en cualquier momento si su precio máximo es inferior al precio spot actual.
* Las instancias MÁS rentables de AWS.
* Útil para las cargas de trabajo que son resistentes a los fallos.

• Trabajos por lotes (Batch Jobs).

• Análisis de datos.

• Procesamiento de imágenes.

• Cualquier carga de trabajo distribuida.

• Cargas de trabajo con una hora de inicio y finalización flexible.

* **No es adecuado para trabajos críticos o bases de datos.**

**Hosts dedicados EC2:**

* Un servidor físico con capacidad de instancia EC2 totalmente dedicado a su uso.
* Permite abordar los requisitos de normativas y utilizar licencias de software vinculadas al servidor existentes (licencias de software por socket, por núcleo, por VM).
* Opciones de compra:

• Bajo demanda - pago por segundo para el host dedicado activo.

• Reservado - 1 o 3 años (sin pago inicial, pago inicial parcial, pago inicial total).

* La opción más cara.
* Útil para el software que tiene un modelo de licencia complicado (BYOL - Bring Your Own License).
* O para empresas que tienen fuertes necesidades de regulación o cumplimiento.

**Instancias dedicadas de EC2:**

* Las instancias se ejecutan en un hardware dedicado para ti.
* Puedes compartir el hardware con otras instancias de la misma cuenta.
* No hay control sobre la ubicación de las instancias (se puede mover el hardware después de la parada/ arranque).

**Reservas de capacidad de EC2:**

* Reserva la capacidad de las instancias bajo demanda en una AZ específica para cualquier duración.
* Siempre tendrás acceso a la capacidad de EC2 cuando la necesites.
* Sin compromiso de tiempo (crear/cancelar en cualquier momento), sin descuentos de facturación.
* Combina con las instancias regionales reservadas y los planes de ahorro para beneficiarte de descuentos en la facturación.
* Se te cobra la tarifa bajo demanda tanto si ejecuta instancias como si no.
* Adecuado para cargas de trabajo ininterrumpidas a corto plazo que necesitan estar en una AZ específica.

**MODELO DE RESPONSABILIDAD COMPARTIDA PARA EC2**

**Para AWS:**

* Infraestructura (seguridad global de la red).
* Aislamiento en hosts físicos.
* Sustitución de hardware defectuoso.
* Validación de la normativa.

**Por parte de nosotros:**

* Reglas de los grupos de seguridad.
* Parches y actualizaciones del sistema operativo.
* Software y utilidades instaladas en la instancia EC2.
* Roles IAM asignados a EC2 y gestión de acceso de usuarios IAM.
* Seguridad de los datos en tu instancia.

**RESUMEN**

* **Instancia EC2:** AMI (SO) + Tamaño de la Instancia (CPU + RAM) + Almacenamiento + Grupos de Seguridad + Datos de Usuario EC2.
* **Grupos de seguridad:** Firewall adjunto a la instancia EC2.
* **Datos de usuario de EC2:** Script lanzado en el primer arranque de una instancia.
* **SSH:** iniciar un terminal en nuestras instancias EC2 (puerto 22).
* **Rol de la Instancia EC2:** enlace a los roles de IAM.
* **Opciones de compra:** On-Demand, Spot, Reservada (Estándar + Convertible + Programada), Host Dedicado, Instancia Dedicada.

**SEC-6: ALMACENAMIENTO DE INSTANCIAS**

**VISIÓN GENERAL DE EBS**

* Un volumen EBS (Elastic Block Store) es una unidad de red que puede adjuntar a las instancias mientras se ejecutan.
* Permite que las instancias persistan los datos, incluso después de su finalización.
* Sólo pueden montarse en una instancia a la vez (a nivel de CCP).
* Están vinculados a una zona de disponibilidad específica.
* Analogía: Piensa en ellos como una "memoria USB de red".
* Nivel gratuito: 30 GB de almacenamiento EBS gratuito de tipo Propósito General (SSD) o Magnético al mes.

**Volumen EBS:**

* Es una unidad de red (es decir, no es una unidad física):

• Utiliza la red para comunicar la instancia, lo que significa que puede haber un poco de latencia.

• Se puede separar de una instancia EC2 y conectarla a otra rápidamente.

* Está bloqueado en una Zona de Disponibilidad (AZ):

• Un volumen EBS en us-east-1a no puede adjuntarse a us-east-1b.

• Para trasladar un volumen, primero hay que hacer un snapshot del mismo.

* Tener una capacidad provisionada (tamaño en GBs, e IOPS):

• Se facturará toda la capacidad aprovisionada.

• Puede aumentar la capacidad de la unidad con el tiempo.

**EBS atributo “Borrar al terminar”:**

* Controla el comportamiento de EBS cuando una instancia EC2 termina.

• Por defecto, se elimina el volumen EBS root / raíz (atributo habilitado).

• Por defecto, cualquier otro volumen EBS adjunto no se elimina (atributo deshabilitado).

* Esto puede ser controlado por la consola de AWS / AWS CLI.
* Caso de uso: preservar el volumen root / raíz cuando se termina la instancia.

**EBS Multi-Attach:**

Los tipos de volumen io1 e io2 pueden adjuntarse a varias instancias.

ESTO VOLUMENES SE ENCUENTRAN FUERA DEL EXAMEN DE CLOUD PRACTITIONER.

Ver más información: <https://docs.aws.amazon.com/es_es/AWSEC2/latest/UserGuide/ebs-volumes-multi.html>

**EBS – PRÁCTICA**

Dentro del servicio **EC2>instancias** elegimos una que tengamos en ejecución y dentro del panel que se nos abre vamos a **almacenamiento** y ahí tendremos la información sobre el volumen.

Si queremos ver todos los volúmenes vamos al menú lateral izquierdo a la sección **Elastic Block Store** y clickeamos en **volúmenes**.

Para agregar un nuevo volumen a una instancia primero debemos saber en que zona esta la instancia ya que los volúmenes están orientados a zonas de disponibilidad.

Por esto en el panel que se nos abre al seleccionar una instancia vamos a la sección **Redes** y buscamos el título **Zona de disponibilidad**.

Ahora para crear el volumen vamos a **Elastic Block Store>volúmenes>crear volumen**. Le damos un tamaño de 2GB, elegimos la zona de disponibilidad en la que se encuentra la instancia y creamos el volumen.

Con el volumen creado clickeamos en su **ID de volumen>Acciones>Asociar volumen** y aquí seleccionamos la instancia y la **asociamos**.

Si eliminamos la instancia el volumen raíz se eliminará, el que asociamos después no. Si queremos modificar esto se hace al crear la instancia en la sección de **almacenamiento**.

**VISIÓN GENERAL DE EBS SNAPSHOTS**

* Haz una copia de seguridad (snapshot) de tu volumen EBS en un momento dado.
* No es necesario separar el volumen para hacer la instantánea, pero se recomienda.
* Puedes copiar las instantáneas a través de AZ (Zona de Disponibilidad) o Región.

**Características:**

Archivo de Snapshots de EBS:

* Mover un snapshot a un "nivel de archivo" que es un 75% más barato.
* La restauración del archivo tarda entre 24 y 72 horas.

Papelera de reciclaje para Snapshots EBS:

* Configura reglas para retener los snapshots eliminados para poder recuperarlos después de un borrado accidental.
* Especifica la retención (de 1 día a 1 año).

**EBS SNAPSHOTS – PRÁCTICA**

En el servicio **EC2** vamos a la sección **Elastic Block Store>Instantáneas>crear instantánea**.

Y desde aquí podemos seleccionar un volumen o una instancia. En este caso vamos a seleccionar un volumen que hayamos creado y le podemos dar un nombre, ahora clickeamos en **crear**.

Ahora nos habrá redirigido a la sección de **instantáneas**, en donde nos aparecerán todas las que hayamos creado.

Si le damos click derecho a la **instantánea** veremos múltiples funciones para ejecutar. Vamos a clickear de estas funciones **copiar instantánea**, lo que nos va a permitir esta función es mover esta información a otras regiones.

Ahora nuevamente en el panel principal de **instantáneas** si clickeamos en su casilla nos abrirá un panel donde podemos ver la información de la misma.

Si queremos pasar esta **instantánea** a un **nivel de archivo**, lo que debemos hacer es seleccionarla, clickear en **Acciones>instantánea de archivo**.

Por último, si queremos **eliminar** la instantánea, la debemos seleccionar y clickear en **Acciones>eliminar instantánea**. Luego para **recuperarla** podemos ir al botón **papelera de reciclaje>recursos**, la seleccionamos y la recuperamos.

**VISIÓN GENERAL DE AMI**

* AMI = Amazon Machine Image.
* Las AMI son una personalización de una instancia EC2:

• Añades tu propio software, configuración, sistema operativo, monitorización...

• Tiempo de arranque/configuración más rápido porque todo el software está preempaquetado.

* Las AMI se construyen para una región específica (y pueden copiarse entre regiones).
* Puedes lanzar instancias EC2 desde:

• Una AMI pública: proporcionada por AWS.

• Tu propia AMI: la creas y la mantienes tú mismo.

• Una AMI de AWS Marketplace: una AMI hecha por otra persona (y potencialmente vendida).

**Proceso de AMI (desde una instancia EC2):**

* Iniciar una instancia EC2 y personalizarla.
* Detener la instancia (para la integridad de los datos).
* Construir una AMI - esto también creará instantáneas de EBS.
* Lanzar instancias desde otras AMIs.

**AMI – PRÁCTICA**

Dentro del servicio **EC2** vamos a crear una **instancia** con el nombre **DemoAMI**, con el inicio rápido de **Amazon Linux**, elegimos un **par de claves**, un **grupo de seguridad** existente, y en la sección **Detalles avanzados>datos de usuario** pegamos las siguientes líneas de código:

#!/bin/bash

# Utiliza esto para tus datos de usuario

# Instala httpd (Version: Linux 2)

yum update -y

yum install -y httpd

systemctl start httpd

systemctl enable httpd

Y creamos la instancia.

Ahora click derecho en la **instancia** y vamos a **imagen y plantillas>crear imagen**.

Le daremos un **nombre** DemoImagen, una **descripción** opcional, y ahora **creamos** la imagen.

A continuación, en el menú lateral izquierdo vamos a la sección **Imágenes>AMI**. Una vez aquí si seleccionamos la **imagen** que creamos, nos dará la opción de crear una instancia a partir de esta AMI, lo que nos permite crear instancias similares de manera más rápida y sin hacer todo el proceso.

Otra manera de crear una instancia con una AMI propia, es desde el panel de crear una nueva instancia, en vez de usar **Amazon Linux**, elegimos **Mis AMI>De mi propiedad** la AMI que creamos, luego configuramos normalmente todo lo demás, y ahora en **Detalles avanzados>datos de usuario** no debemos copiar las líneas de código que hicimos en la instancia creada con anterioridad, solamente poner las primeras 3 para que todo se instale correctamente:

#!/bin/bash

# Utiliza esto para tus datos de usuario

# Instala httpd (Version: Linux 2)

echo "<h1>Hola Mundo desde $(hostname -f)</h1>" > /var/www/html/index.html

**VISIÓN GENERAL DE EC2 IMAGE BUILDER**

* Se utiliza para automatizar la creación de máquinas virtuales o imágenes de contenedores.
* => Automatizar la creación, mantener, validar y probar las AMIs de EC2.
* Puede ejecutarse de forma programada (semanalmente, cada vez que se actualizan los paquetes, etc.…).
* Servicio gratuito (sólo se paga por los recursos subyacentes).

**Cómo son los pasos:**

1. El EC2 Image Builder crea la instancia.
2. Instancia del constructor: se encarga de construir componentes aplicados (personaliza el software en la instancia).
3. Ahora se crea una nueva AMI.
4. Esta AMI va a pasar a la instancia de EC2 de prueba. Testeando que todo funcione correctamente. Se ejecuta el conjunto de pruebas.
5. Finalmente, si todo está bien el AMI es distribuido (pueden ser en varias regiones).

**EC2 IMAGE BUILDER: CREACION DE UN PIPELINE/CANALIZACIÓN – PRÁCTICA**

Vamos al servicio **EC2 Image Builder** y una vez allí adentro vamos a **crear canalización de imágenes**.

**Pasos a seguir:**

1. Le damos un **nombre** como MiDemoPipeline, luego en **Programación de compilación** vamos a utilizar la **manual** y pasamos al siguiente paso.
2. Elegir receta: usaremos **crear nueva receta**, en donde el tipo de imagen sea **Imagen de Amazon Machine**. Le daremos un **nombre** como MiDemoReceta con la versión 1.0.0 ya que es la primera que creamos. En **Imagen base** utilizamos **Seleccionar imágenes administradas** y elegimos **Amazon Linux**, y además utilizamos el **inicio rápido**, en el título **nombre de la imagen** vamos a utilizar **Amazon Linux 2 x86** (esto es porque estamos usando T2micro que es la gratuita). En la sección de **Componentes** buscamos el componente **amazon-corretto-11-headless** para tener la versión 11 de Java, y **aws-cli-version-2-linux**. Damos en siguiente.
3. Definir la configuración de la infraestructura: elegimos **crear una nueva configuración de infraestructura**. Le damos un **nombre** como MiDemoInfra, y ahora nos va a pedir un **rol de IAM** el cual deberá contener los siguientes permisos: **EC2InstanceProfileForImageBuilder**, **EC2InstanceProfileForImageBuilderECRContainerBuilds** y **AmazonSSMManagedInstanceCore**. Con el nombre del rol **EC2InstanceProfileForImageBuilder**. Ahora en la sección **Infraestructura de AWS** elegimos la **T2.micro**. Damos en siguiente.
4. Definir la configuración de la distribución: elegimos la primera opción con los valores predeterminados y damos en siguiente.
5. Revisar: revisamos que todo este bien y creamos el pipeline.

**IMAGE BUILDER: EJECUCIÓN DE UN PIPELINE - PRÁCTICA**

**Pasos a seguir:**

1. Clickeamos el pipeline creado.
2. En **Acciones>Ejecutar canalización**.
3. Vamos al servicio de **EC2>Instancias** y nos aparecerá la instancia en ejecución con el nombre **Build Instance**, pero esta estará en el cuadro de comprobación en **inicializando**.
4. Una vez que pase de estado la imagen, en las instancias aparecerá una con el nombre **Test Instance for nombreDeLaReceta** y la instancia de Build instance habrá finalizado.
5. Una vez terminado todo el proceso de construcción vamos a **lanzar una instancia**, elegimos de **Mis AMI**, **MiDemoReceta** que fue la que creamos en el video anterior, seleccionamos un **grupo de seguridad** existente, un **par de claves** existentes y ahora en **Detalles avanzados** no vamos a introducir NADA.
6. Lanzamos la instancia.
7. Nos conectamos a la instancia con el nombre de usuario ec2-user.

**EC2 INSTANCE STORE**

* Los volúmenes EBS son unidades de red con un rendimiento bueno, pero “limitado".
* Si necesitas un disco de hardware de alto rendimiento, utilizas EC2 Instance Store.

**Características:**

* Mejor rendimiento de Entrada/Salida.
* Los almacenes de instancias EC2 pierden su almacenamiento si se detienen (son efímeros).
* Bueno para el buffer / cache / datos de memoria virtual / contenido temporal.
* Riesgo de pérdida de datos si el hardware falla.
* Las copias de seguridad y la replicación son responsabilidad tuya.

**VISIÓN GENERAL DE EFS – ELASTIC FILE SYSTEM**

* NFS (network file system / sistema de archivos de red) gestionado que puede montarse en 100 EC2s.
* EFS funciona con instancias EC2 de Linux en multi-AZ.
* Alta disponibilidad, escalable, es más caro (3x gp2), pago por uso, sin planificación de capacidad.
* Varios grupos de instancias pueden están conectados al EFS.

**EFS INFREQUENT ACCESS:**

* Clase de almacenamiento con costes optimizados para los archivos a los que no se accedes a diario.
* Hasta un 92% menos de coste en comparación con EFS Standard.
* EFS moverá automáticamente tus archivos a EFS-IA basándose en la última vez que se accedió a ellos.
* Habilita EFS-IA con una política de ciclo de vida (Lifecycle Policy).
* Ejemplo: mover a EFS-IA los archivos a los que no se ha accedido en 60 días.
* Transparente para las aplicaciones que acceden a EFS.

**MODELO DE RESPONSABILIDAD COMPARTIDA PARA EL ALMACENAMIENTO DE EC2**

**Por parte de AWS:**

* Infraestructura.
* Replicación de datos para volúmenes EBS y unidades EFS.
* Sustitución de hardware defectuoso.
* Asegurar que sus empleados no puedan acceder a tus datos.

**Por parte del cliente:**

* Configuración de procedimientos de copia de seguridad / instantánea.
* Configuración de la encriptación de datos.
* Responsabilidad de los datos en las unidades.
* Comprender el riesgo de utilizar EC2 Instance Store.

**VISIÓN GENERAL DE AMAZON FSx**

* Lanzar sistemas de archivos de alto rendimiento de terceros en AWS.
* Servicio totalmente gestionado.

Hay 3 tipos:

* FSx para Lustre.
* FSx para Windows File Server.
* FSx para NetApp ONTAP.

**Amazon FSx para Windows File Server:**

* Un sistema de archivos compartido nativo de Windows totalmente gestionado, altamente fiable y escalable.
* Construido sobre Windows File Server.
* Soporta el protocolo SMB y Windows NTFS.
* Integrado con Microsoft Active Directory.
* Se puede acceder desde AWS o desde tu infraestructura local.

**Amazon FSx para Lustre**:

* Un almacenamiento de archivos totalmente gestionado, de alto rendimiento y escalable para High Performance Computing (HPC).
* El nombre Lustre deriva de "Linux" y "cluster".
* Machine Learning, análisis, procesamiento de vídeo, modelado financiero, ...
* Escala hasta 100s GB/s, millones de IOPS, latencias sub-ms.

**RESUMEN**

**Volúmenes EBS:**

* Unidades de red adjuntas a una instancia EC2 a la vez.
* Asignados a una zona de disponibilidad.
* Puede utilizar EBS Snapshots para copias de seguridad / transferir volúmenes EBS a través de AZ.

**AMI:** crea instancias EC2 listas para usar con nuestras personalizaciones.

**EC2 Image Builder:** construye, prueba y distribuye automáticamente AMIs.

**EC2 Instance Store:**

* Disco de hardware de alto rendimiento unido a nuestra instancia EC2.
* Se pierde si nuestra instancia se detiene / termina.

**EFS:** sistema de archivos en red, se puede adjuntar a 100s de instancias en una región.

**EFS-IA:** clase de almacenamiento de coste optimizado para archivos de acceso poco frecuente.

**FSx para Windows:** sistema de archivos en red para servidores Windows.

**FSx para Lustre:** sistema de archivos Linux de alto rendimiento informático.

**SEC-7: ELB y ASG – ELASTIC LOAD BALANCING y AUTO SCALING GROUPS**

**ALTA DISPONIBILIDAD, ESCALABILIDAD Y ELASTICIDAD**

**Escalabilidad y alta disponibilidad:**

* La escalabilidad significa que una aplicación/sistema puede manejar mayores cargas adaptándose.
* Hay dos tipos de escalabilidad:

• Escalabilidad vertical.

• Escalabilidad horizontal (= elasticidad).

* La escalabilidad está vinculada, pero es diferente a la alta disponibilidad.

**Escalabilidad vertical:**

* La escalabilidad vertical significa aumentar el tamaño de la instancia.
* Por ejemplo, tu aplicación se ejecuta en una instancia t2.micro.
* Escalar esa aplicación verticalmente significa ejecutarla en una instancia t2.large.
* La escalabilidad vertical es muy común para sistemas no distribuidos, como una base de datos.
* Por lo general, hay un límite en cuanto a lo que se puede escalar verticalmente (límite de hardware).

**Escalabilidad horizontal:**

* Escalabilidad horizontal significa aumentar el número de instancias / sistemas para la aplicación.
* El escalado horizontal implica sistemas distribuidos.
* Esto es muy común para las aplicaciones web / aplicaciones modernas.
* Es fácil escalar horizontalmente gracias a las ofertas en el Cloud como Amazon EC2.

**Alta disponibilidad:**

* La alta disponibilidad suele ir de la mano del escalado horizontal.
* Alta disponibilidad significa ejecutar la aplicación / sistema en al menos 2 zonas de disponibilidad.
* El objetivo de la alta disponibilidad es sobrevivir a la pérdida del centro de datos (desastre).

**Alta disponibilidad y escalabilidad para EC2:**

* Escalado vertical: Aumentar el tamaño de la instancia (= escalar hacia arriba / abajo).

• Desde: t2.nano - 0.5G de RAM, 1 vCPU.

• A: u-12tb1.metal - 12,3 TB de RAM, 448 vCPUs.

* Escalado horizontal: Aumentar el número de instancias (= escalado hacia fuera / hacia dentro).

• Auto Scaling Group.

• Load Balancer.

* Alta disponibilidad: Ejecutar instancias para la misma aplicación a través de múltiples AZ.

• Auto Scaling Group multi AZ.

• Load Balancer multi AZ.

**Escalabilidad vs Elasticidad (vs Agilidad)**

Escalabilidad: capacidad de acomodar una mayor carga reforzando el hardware (scale up), o añadiendo nodos (scale out).

Elasticidad: una vez que un sistema es escalable, la elasticidad significa que habrá cierto "autoescalado" para que el sistema pueda escalar en función de la carga. Esto es "amigable con el Cloud": pago por uso, adecuación a la demanda, optimización de costes.

Agilidad: (no relacionado con la escalabilidad - distractor) los nuevos recursos de IT están a un clic de distancia, lo que significa que se reduce el tiempo para poner esos recursos a disposición de los desarrolladores de semanas a sólo minutos.

**ELASTIC LOAD BALANCING (ELB)**

Los Load Balancers (equilibradores de carga) son servidores que reenvían el tráfico de Internet a múltiples servidores (Instancias EC2) en sentido descendente.

**¿Por qué utilizar un Load Balancer?**

* Distribuir la carga entre múltiples instancias descendentes.
* Exponer un único punto de acceso (DNS) en tu aplicación.
* Manejar sin problemas los fallos de las instancias descendentes.
* Realiza comprobaciones periódicas del estado de tus instancias.
* Proporcionar terminación SSL (HTTPS) para tus sitios web.
* Alta disponibilidad entre zonas.

**¿Por qué utilizar un Elastic Load Balancer (ELB)?**

* Un ELB (Elastic Load Balancer) es un Load Balancer / Equilibrador de Carga gestionado.

• AWS garantiza su funcionamiento.

• AWS se encarga de las actualizaciones, el mantenimiento y la alta disponibilidad.

• AWS sólo proporciona unos pocos controles de configuración.

* Cuesta menos configurar tu propio Load Balancer pero te supondrá mucho más esfuerzo (mantenimiento, integraciones).
* 3 tipos de Load Balancer ofrecidos por AWS:

• Load Balancer de aplicaciones (sólo HTTP / HTTPS) - Capa 7.

• Load Balancer de red (rendimiento ultra alto, permite TCP) - Capa 4.

• Load Balancer clásico - Capa 4 y 7.

**APPLICATION LOAD BALANCER (ALB) – PRÁCTICA**

En el servicio **EC2** lanzamos dos instancias con el nombre **Primer Servidor** y **Segundo Servidor** con las configuraciones que ya veníamos haciendo normalmente. Estas dos instancias son las que el **Load Balancer** va a controlar para distribuir la carga.

Una vez creadas las instancias nos dirigimos en el menú lateral izquierdo a la sección **Equilibrio de carga>Balanceadores de carga>Create load balancer**. De los 3 tipos que hay vamos a elegir **Application Load Balancer**.

Ahora le damos un **nombre** como DemoALB, el **esquema** que sea **internet-facing**, el **IP** que sea **IPv4**.

En la sección **Network mapping** el VPC lo dejamos por defecto y en **mappings** vamos a seleccionar todas las zonas de disponibilidad.

En la sección **Security Groups** vamos a crear un nuevo grupo de seguridad, al cual le daremos un **nombre**, una **descripción**, y en **reglas de entrada** vamos a buscar **HTTP** con el **origen** en **anywhere**. Y ahora creamos el grupo de seguridad. Con esto listo ahora elegimos este grupo de seguridad recién creado y podemos sacar el que estaba por default.

Ahora el paso en la sección **Listener and routing** es verdaderamente importante porque es donde voy a configurar las instancias donde se van a reenviar el tráfico del **load balancer**.

Vamos a clickear en **create target group**, y una vez allí adentro vamos a seleccionar **instancias**, le damos un **nombre**, dejamos el resto por como esta por defecto y vamos a **siguiente**. Ahora seleccionamos las **instancias** que en este caso son Primer Servidor y Segundo Servidor y clickeamos en **Include as pending below**, y creamos el grupo.

Ahora volvemos a **Listener and routing** y seleccionamos este grupo creado.

Por último, clickeamos en **Create load balancer**.

A continuación, nos redirige a la ventana donde aparece nuestro load balancer, vamos a seleccionarlo y copiar el **DNS name** y pegarlo en el navegador, vamos a ver que si lo recargamos el IP cambia porque es uno de los dos servidores que elegimos el que nos da la respuesta.

**VISIÓN GENERAL DE AUTO SCALING GROUPS (ASG)**

* En la vida real, la carga de tus sitios web y aplicaciones puede cambiar.
* En el Cloud, puedes crear y deshacerte de servidores muy rápidamente.
* El objetivo de un Auto Scaling Group (ASG) es:

• Reducir (añadir instancias de EC2) para adaptarse a un aumento de la carga.

• Aumentar (eliminar instancias EC2) para que coincida con una disminución de la carga.

• Asegurar que tenemos un número mínimo y máximo de máquinas en funcionamiento.

• Registrar automáticamente nuevas instancias en un Load Balancer.

• Reemplazar las instancias en mal estado.

* Ahorro de costes: sólo se ejecuta a una capacidad óptima (principio del Cloud).

**AUTO SCALING GROUPS – PRÁCTICA**

Dentro del servicio **EC2** en el menú lateral izquierdo vamos a la sección **Auto Scaling>Grupos de Auto scaling>Crear grupo de auto scaling**.

* Paso 1:

Elegir un **nombre** como DemoASG, y a continuación vamos a tener que crear una **plantilla de lanzamiento**.

Esta plantilla llevará un **nombre**, un inicio rápido con **Amazon Linux**, tipo de instancia de **t2.micro**, no hace falta insertar un **par de claves**, en grupos de seguridad elegimos uno existente como el **launch-wizard-1**, ahora en **detalles avanzados>datos de usuario** utilizamos las mismas líneas de código de siempre:

#!/bin/bash

# Utiliza esto para tus datos de usuario

# Instala httpd (Version: Linux 2)

yum update -y

yum install -y httpd

systemctl start httpd

systemctl enable httpd

echo "<h1>Hola Mundo desde $(hostname -f)</h1>" > /var/www/html/index.html

Y creamos la plantilla de lanzamiento.

Ahora la elegimos y clickeamos en **siguiente**.

* Paso 2:

Elegimos las zonas de disponibilidad, en este ejemplo seleccionamos todas.

Click en **siguiente**.

* Paso 3:

Debemos elegir un **Balance de carga**, como tenemos uno existente por el video anterior, elegimos la opción **asociar a un balanceador de carga existente**, y elegimos el que creamos con anterioridad.

En **comprobaciones de estado** activamos el **ELB**.

Click en **siguiente**.

* Paso 4:

En **tamaño del grupo** en este ejemplo vamos a utilizar en **capacidad deseada** = 2, **capacidad mínima** = 1, **capacidad máxima** = 4.

En **Políticas de escalado** lo dejamos en **ninguno**.

Click en **siguiente**.

* Paso 5 y 6:

Podemos agregar si queremos notificaciones y etiquetas, en este ejemplo no las usamos.

* Paso 7:

Revisamos y creamos.

**ESTRATEGIAS DE AUTO SCALING GROUPS (ASG)**

Hay varios tipos:

**Escalado Manual:** actualizar el tamaño de un ASG manualmente.

**Escalado dinámico:** Responde a los cambios en la demanda.

* Escalado simple: Cuando se activa una alarma de CloudWatch (por ejemplo, CPU > 70%), se añaden 2 unidades o también si baja el rendimiento de la CPU < 30% se quita una unidad.
* Escalado de seguimiento de objetivos: por ejemplo, quiero que la media de la CPU de ASG se mantenga en torno al 40%.
* Escalado programado: Anticipar un escalado basado en patrones de uso conocidos. Ejemplo: aumentar la capacidad mínima a 10 a las 17 horas de los viernes.

**Escalado predictivo:**

* Utiliza el Machine Learning para predecir el tráfico futuro con antelación.
* Aprovisiona automáticamente el número correcto de instancias EC2 por adelantado.
* Útil cuando tu carga tiene patrones predecibles basados en el tiempo.

**RESUMEN – ELB y ASG**

* Alta disponibilidad vs escalabilidad (vertical y horizontal) vs elasticidad vs agilidad en el Cloud.
* Elastic Load Balancers (ELB):

• Distribuyen el tráfico entre las instancias EC2 del backend, pueden ser Multi-AZ.

• Soporta chequeos de salud.

• 3 tipos: Application LB (HTTP - L7), Network LB (TCP - L4), Classic LB (antiguo).

* Auto Scaling Groups (ASG):

• Implementa la elasticidad para tu aplicación, a través de múltiples AZ.

• Escala las instancias EC2 en función de la demanda de tu sistema, sustituye las instancias en mal estado.

• Integrado con el ELB.

**SEC-8: S3**

**VISIÓN GENERAL DE S3**

* Amazon S3 es uno de los principales bloques de construcción de AWS.
* Se anuncia como almacenamiento de "escala infinita".
* Muchos sitios web utilizan Amazon S3 como columna vertebral.
* Muchos servicios de AWS utilizan Amazon S3 como una integración también.
* Tendremos una aproximación paso a paso a S3.
* El examen CCP requiere un conocimiento "más profundo" sobre S3.

**S3 Casos de uso:**

* Copia de seguridad y almacenamiento.
* Recuperación de desastres.
* Almacenamiento en el Cloud híbrido.
* Alojamiento de aplicaciones.
* Alojamiento de medios.
* Data Lakes y análisis de big data.
* Entrega de software.
* Sitio web estático.

**Visión general de S3 – Buckets:**

* Amazon S3 permite almacenar objetos (archivos) en "buckets" (directorios).
* Los buckets deben tener un nombre único a nivel global (en todas las regiones, todas las cuentas).
* Los buckets se definen a nivel de región.
* S3 parece un servicio global pero los buckets se crean en una región.

• Convención de nombres.

• Sin mayúsculas.

• Sin guiones bajos.

• 3-63 caracteres de longitud.

• No es una IP.

• Debe comenzar con una letra minúscula o un número.

**Vision general de S3 – Objetos:**

* Los objetos (archivos) tienen una clave.
* La clave es la ruta COMPLETA:

• s3://mi-bucket/mi-archivo.txt

• s3://mi-bucket/mi\_carpeta1/otra\_carpeta/mi\_archivo.txt.

* La clave se compone de prefijo + nombre del objeto:

• s3://mi-bucket/mi\_carpeta1/otra\_carpeta/mi\_fichero.txt

* No existe el concepto de "directorios" dentro de los buckets (aunque la interfaz de usuario te hará pensar lo contrario).
* Sólo claves con nombres muy largos que contienen barras inclinadas (“/").
* Los valores de los objetos son el contenido del cuerpo:

• El tamaño máximo del objeto es de 5TB (5000GB).

• Si se suben más de 5GB, se debe usar "carga de varias partes".

* Metadatos (lista de pares clave/valor de texto - metadatos del sistema o del usuario).
* Etiquetas (par clave/valor Unicode - hasta 10) - útil para la seguridad/ciclo de vida.
* ID de la versión (si el versionado está activado).

**S3 – PRÁCTICA**

Buscamos el servicio **S3** y clickeamos en **Crear bucket**.

Le daremos un **nombre** que tiene que ser único, y dejaremos el resto por defecto en este ejemplo.

Una vez creado clickeamos en el nombre del bucket, y allí adentro podremos **cargar** archivos, si subimos uno luego podemos clickear en él y nos redirigirá a una pestaña con toda la información del archivo, donde encontraremos por ejemplo la **url** del archivo, pero ATENCIÓN para que esa url funcione debemos haber puesto con anterioridad a la hora de crear el bucket que sea de **origen público**.

También dentro de la pestaña del bucket podemos **crear una carpeta** y cargar archivos dentro de la misma.

**SEGURIDAD EN S3: POLÍTICA DE BUCKETS**

**Basado en el usuario:**

* Políticas de IAM: qué llamadas a la API deben permitirse para un usuario específico desde la consola de IAM.

**Basadas en recursos:**

* Políticas de bucket - reglas a nivel de bucket desde la consola de S3 - permite cuentas cruzadas.
* Lista de control de acceso a objetos (ACL).
* Lista de control de acceso a buckets (ACL)

**Nota:** un usuario de IAM puede acceder a un objeto de S3 si:

* Los permisos IAM del usuario lo permiten O la política de recursos lo PERMITE.
* Y no hay una DENEGACIÓN explícita.

**Encriptación:** encripta los objetos en Amazon S3 utilizando claves de encriptación.

**Ejemplos:**

* Acceso público: el visitante anónimo puede acceder al bucket mediante políticas de seguridad el bucket que permite el acceso público.
* Acceso del usuario a S3: El usuario IAM con sus políticas puede acceder al bucket.
* Acceso de la instancia EC2: la instancia accede al bucket mediante permisos IAM.
* Acceso entre cuentas: Un usuario IAM de otra cuenta de AWS accede al bucket mediante políticas de seguridad del bucket que permite las cuentas cruzadas.

**Políticas de bucket S3:**

Políticas basadas en JSON:

* Resources: buckets y objetos.
* Actions: Conjunto de API para permitir o denegar.
* Effect: Permitir / Denegar.
* Principal: La cuenta o usuario al que aplicar la política.

Usar el bucket S3 para aplicar la política:

* Conceder acceso público al bucket.
* Forzar que los objetos se cifren al subirlos.
* Conceder acceso a otra cuenta (cuenta cruzada).

**SEGURIDAD EN S3: POLÍTICAS DE BUCKETS – PRÁCTICA**

En el servicio **S3** entramos al bucket que creamos en el video anterior y vamos a la sección **Permisos**. Una vez dentro vamos a deshabilitar las funciones de **bloquear acceso público**, así ahora podemos agregar políticas de bucket.

Ahora en la sección **Políticas de bucket** seleccionamos **editar**, la cual nos redirigirá a otra ventana. Una vez allí adentro vamos a **generador de políticas**, el cual nos abre una nueva pestaña, y es aquí donde vamos a crear nuestras políticas para el bucket.

Paso 1: seleccionamos el tipo de política **S3 Bucket Policy**.

Paso 2: en **principal** agregamos el **\*** para que pueda acceder cualquier usuario, en **actions** seleccionamos **GetObject**, acto seguido necesito el **ARN**, el cual está en la sección **Políticas de bucket>editar**, una vez que lo peguemos vamos a agregarle a ese ARN una barra y el asterisco (**/\***) para poder acceder a todos los objetos.

Click en **Add statement**

Paso 3: click en **Generate Policy**. Esto nos va dar la política en formato JSON. La copiamos, volvemos a la pestaña de **Políticas de bucket**, la pegamos y **guardamos cambios**.

**VISIÓN GENERAL DE SITIOS WEB USANDO S3**

* S3 puede alojar sitios web estáticos y tenerlos accesibles en la www
* La URL del sitio web será:

• <nombre-del-bucket>.s3-website-<región-AWS>.amazonaws.com O bien:

• <nombre-del-bucket>.s3-website.<región-AWS>.amazonaws.com

* Si obtienes un error 403 (Prohibido), ¡asegúrate de que la política del bucket permite las lecturas!

**SITIO WEB USANDO S3 – PRÁCTICA**

Ingresamos al servicio **S3** y en el bucket ya creado vamos a agregar las dos imágenes y el archivo **index** que nos dio como material el curso.

Una vez cargadas, nos vamos a la sección **Propiedades>Alojamiento de sitios web estáticos>editar**.

Clickeamos en **habilitar**, en **documento de índice** escribimos el nombre del archivo que subimos recién, es decir, **index.html** y guardamos cambios.

Ahora vamos a poder acceder a esta página.

Por último, dentor del bucket volvemos a **Propiedades>Alojamiento de sitios web estáticos** y allí tendremos la **url**.

**VISIÓN GENERAL DEL VERSIONADO DE S3**

* Puedes versionar tus archivos en Amazon S3.
* Se activa a nivel de bucket.
* La misma clave de sobrescritura incrementará la "versión": 1, 2, 3....
* Es una buena práctica versionar tus buckets:

• Protege contra los borrados involuntarios (posibilidad de restaurar una versión).

• Facilidad para volver a la versión anterior.

* **Notas**:

• Cualquier archivo que no esté versionado antes de activar el versionado tendrá la versión "nula".

• Suspender el versionado no elimina las versiones anteriores.

**VERSIONADO DE S3 – PRÁCTICA**

Entramos al servicio **S3**, luego al bucket que ya creamos, y vamos a la sección **Propiedades>Control de versiones de buckets>editar**, seleccionamos **habilitar** y guardamos cambios.

Ahora tenemos activo el versionado, lo que significa que si por ejemplo editamos algo del archivo **index.html** que subimos en el video anterior, y volvemos a subir este archivo a S3, no tendremos dos index.html, sino que se actualiza.

Además, podemos entrar en la sección **Objetos** y ver los archivos que tenemos, y cliquear en **Mostrar versiones** para ver cuando se hicieron las modificaciones y ver el archivo antes de su modificación.

**REGISTRO DE ACCESO AL SERVIDOR S3**