Hatim Kaddouch Mrini

CONFIGURACIÓN DE UN ENTORNO CLOUD SEGURO Y DE ALTA DISPONIBILIDAD EN AMAZON WEB SERVICES (AWS)

TRABAJO DE FINAL DE GRADO

Dirigido por el Dr. Jordi Castellà Roca

Grado de Ingeniería Informática

UNIVERSITAT ROVIRA I VIRGILI

Tarragona

2021

Resumen

Con el paso del tiempo más empresas optan por realizar sus servicios de forma remota mediante Internet. Esto ha provocado que se tengan que buscar nuevas arquitecturas para sus sistemas que puedan evitar pérdidas, ya que al ser accesibles de forma remota se encuentran muy expuestos a posibles fallos, ya sean de seguridad o del mismo sistema. Para ello, se usan diferentes proveedores de *Cloud Computing*, de los cuales, nosotros nos centraremos en *Amazon Web Services*.

Mediante *AWS* se recreará un prototipo de sistema seguro que tratará de menguar o eliminar los problemas actuales de los servidores, como serían la escalabilidad, la tolerancia al fallo y la seguridad.

Resum

Amb el pas del temps, més empreses opten per realitzar els seus serveis de forma remota mitjançant Internet. Això ha provocat la busca de noves arquitectures per els seus sistemes capaces d'evitar pèrdues, ja que al ser accessibles de forma remota es troben altament exposats a possibles fallades de seguretat o del mateix sistema. Per això s'usen diferents proveïdors de *Cloud Computing*, dels quals ens centrarem en *Amazon Web Services*.

Mitjançant *AWS* recrearem un prototip de sistema segur capaç de disminuir o eliminar els problemes actuals dels servidors, como serien l'escalabilitat, la tolerància a errors i la seguretat.

Abstract

Over time, more companies choose to carry out their services remotely over the Internet. This has meant that they must find new architectures for their systems that can avoid losses, since being remotely accessible they are highly exposed to possible failures, either security or the system itself. To avoid it, there are different Cloud Computing providers that can be used, of which we Will focus on Amazon Web Services.

Using AWS, we will be able to recreate a prototype of a secure system that tries to reduce or eliminate current problems with servers, such as scalability, fault tolerances and security.

Contenido

Contenido

1. Introducción

Internet ha sido y es el medio de información más rápido usado por las personas (1), cobrando éste una gran importancia en la era en la que nos encontramos. Al igual que en una casa tenemos los servicios básicos (agua, luz, gas, etc..) disponer de una conexión a Internet también se ha convertido en un elemento esencial en nuestras vidas. La pandemia del Covid-19 ha incrementado esta dependencia (2), permitiendo que en algunos sectores se pudiera seguir con la actividad. Por ejemplo, las clases se han realizado online, se ha implantado el teletrabajado, y ha habido un incremento en las compras por Internet, etc.

No obstante, la conectividad (Internet) no es suficiente. Los servicios que se ofrecen a través de la red también se deben de adaptar, o más concretamente las arquitecturas de los servicios prestados por las empresas se deben modificar para proporcionar esta actividad. En este sentido el *Cloud Computing*, de manera simplificada, es una tecnología que permite tener acceso a servicios de forma remota, almacenamiento de datos y procesamiento de estos a través de Internet. Esto nos ha permitido solucionar o menguar los siguientes problemas que se han encontrado en los sistemas convencionales. Sin embargo, los servicios en el Cloud deben hacer frente a las siguientes situaciones:

- Sobrecarga de los servidores debido a la gran cantidad de gente intentando acceder a ellos. Esto se ha observado mucho en los últimos años, y es que los sistemas tienen poco aguante a accesos en masa por parte de la gente.
- La tolerancia a los fallos siempre ha sido un problema, ya que los servidores se encontraban limitados a un número limitado de máquinas. Con *Cloud Computing* se puede usar la gran potencia de la nube para tener más servidores en el caso de que falle alguno.
- La seguridad es un aspecto indispensable en los sistemas en la nube, ya que son accesibles de forma remota, por lo que se encuentran altamente expuestos.

1.1. Objetivos

El objetivo del trabajo es conocer y aprender sobre el funcionamiento de *Amazon Web Services* (4) donde se configurará un entorno que tendrán en cuenta los tres puntos a trabajar mencionados en el apartado anterior.

En este entorno se intentará balancear la carga de trabajo de los diferentes servidores de tal manera que no se sobrecarguen y fallen. Esto se conseguirá hacer usando los elementos propios del sistema *Cloud* que se explicarán más adelante. Aparte de este balanceo de

carga, tendremos en cuenta una alta escalabilidad manual, es decir, el administrador del sistema podrá incluir todos los servidores que desee manualmente.

El segundo punto importante a tener en cuenta es la redundancia de la que dispondrá, usándose esta para el punto anterior y para maximizar la tolerancia a los fallos. En este aspecto nos aseguraremos de que, cuando falle un servidor, podamos redirigir la información a otro idéntico. Con esta característica también conseguimos hacer que el sistema resiliente, es decir, un sistema capaz de recuperarse de una falla.

Finalmente, el último aspecto que tendremos en cuenta es la seguridad de nuestro sistema, configurando nuestro cortafuegos y creando una DMZ para poder mantener toda nuestra arquitectura aislada de posibles ataques. También se configurarán varias subredes para menguar los daños de un ataque, ya que se tendrá que acceder a cada subred por separado, aumentando la complejidad para acceder a todos nuestros servidores.

Con todo esto, conseguiremos montar un prototipo de sistema seguro que se puede expandir tanto como queramos.

1.2. Justificación/Motivación

En el caso que una empresa decida ofrecer sus servicios a través de Internet, debería implantar en el Cloud una arquitectura que cumpliera con los puntos mencionados. Si tiene una incidencia en el servicio, fallo de seguridad, estará perdiendo dinero, llegando al caso extremo de verse obligada a finalizar su actividad.

Actualmente disponemos de un mercado con una gran diversidad de herramientas que nos facilitan la implementación de arquitecturas que cumplan con los objetivos mencionados. También tenemos un gran número de cursos para estas herramientas con certificados oficiales que permiten la formación de los usuarios.

1.3.Organización de la memoria

En la Sección 2, se describen las tecnologías o conceptos utilizados en el trabajo. La Sección 3 incluye una explicación del entorno AWS, donde se entra en detalle en sus servicios usados para realizar este sistema seguro. La Sección 4 ya se introduce la preparación del entorno, el diseño de la arquitectura y, seguidamente, la implementación de esta con la comprobación de su correcta ejecución. Finalmente, la Sección 5 encontramos las conclusiones y el futuro trabajo que se podría realizar sobre este sistema.

2. Tecnologías

Para realizar el entorno seguro de este trabajo se necesitan varios elementos básicos de las redes de comunicaciones. En este apartado, se hará una introducción a estos elementos y, más adelante, se entrará más en profundidad en ellos especificando las características propias en *Amazon Web Services*.

Redes

En primer lugar, se describen las redes de comunicaciones y sus elementos.

Una **red** es un conjunto de dispositivos que se encuentran conectados unos con los otros, con la finalidad de poder intercambiar información y compartir sus recursos. Esta red puede estar formada por **subredes**, con la diferencia de que estas tienen un rango de direcciones lógicas menor a las redes.

En una red típica es común encontrar una **zona desmilitarizada** (DMZ). Esta zona recibe ese nombre por ser una red que se encuentra entre nuestra red principal y una red externa, regularmente, Internet. Esta red se usa para añadir seguridad al sistema, ya que permite que la red interna y la red externa se conecten a ella, pero deniega las conexiones desde la DMZ a la red interna. Es decir, hace de muro entre la red externa y la red interna para evitar accesos innecesarios a la red interna.

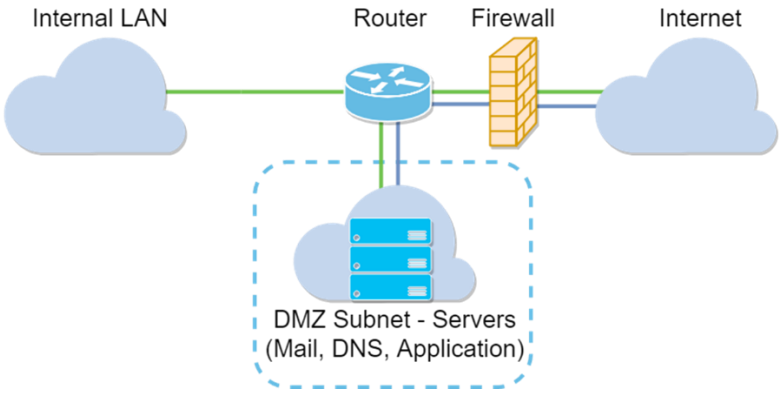

Figura 2.1 DMZ

El **servidor** es una máquina capaz de recibir peticiones externas y devolver una respuesta a esta. Estas máquinas pueden ser muy vulnerables puesto que son accesibles remotamente y se encuentran expuestas a distintos ataques. Para ello se usan varios elementos que impiden el acceso indeseado.

Seguridad

Un elemento de seguridad muy usado en todos los sistemas es el **cortafuegos**. Los cortafuegos son una parte de nuestro sistema por donde pasan todos los mensajes antes de entrada o salida de nuestra red privada, siendo analizados por este. Si alguno de estos mensajes no cumple con los criterios establecidos en el cortafuegos, se rechaza.

Una vez pasado este primer análisis, encontramos un elemento específico para cada servidor web, el ***Web Application Firewall*** (WAF). Este hace un análisis parecido al cortafuegos tradicional, pero ya dentro de cada máquina y a nivel de aplicación, principal diferencia con el cortafuegos. Realiza un examen a todas las peticiones antes de que lleguen a la aplicación y permite el paso a las que cumplan con todas sus reglas.

Finalmente, encontramos un último elemento de seguridad, el **proxy**. Este es un servidor que hace de intermediario entre el usuario y el servidor, recibiendo la petición y la respuesta por ambas partes y redireccionándolas él mismo. Con esto se consigue otra capa de seguridad añadiendo más control y registro del tráfico.

Enrutamiento

El elemento más importante a la hora de enrutar es el **router**¸ el dispositivo que permite la interconexión entre las diferentes redes e instancias.

La **tabla de enrutamiento** es un componente donde se registran todas las rutas que han de seguir los paquetes a través de los nodos que disponemos en nuestra red. Este contiene todas las reglas para cualquier tipo de paquete desde que se recibe hasta que llega a su destino.

El último elemento al que vamos a hacer mención el **balanceador de carga**, que se ocupará de redistribuir los paquetes entre los distintos servidores. Esta redistribución se hace para evitar saturación en los servidores, minimizar tiempos de respuesta y mejorar el rendimiento de los servicios.

Otros

Las ***Gateways*** son puertas de enlace entre dispositivos, sistemas o servicios, como podrían ser Internet u otra instancia.

Los ***Endpoints*** son los puntos de enlace de los servicios, las URLs del punto de entrada de un servicio web de AWS.

3. Amazon Web Services

Amazon Web Services, comúnmente conocido como AWS, es, como indica su nombre, la plataforma *Cloud* de Amazon. Es el sistema *Cloud* más usado y distribuido alrededor del mundo (3) por varias razones; dispone de varias certificaciones y auditorias de seguridad que hacen que sea 100% fiable; permite acceso a distintos tipos de bases de datos; ha logrado reducir sus costos hasta el punto de ser de los más bajos del mercado; es un sistema resiliente, entre muchas otras ventajas.

En AWS encontramos una gran cantidad de servicios, de los cuales, nos centraremos en dos, llamados ***VPC*** y ***EC2***.

3.1. Virtual Private Cloud

El servicio de ***Virtual Private Cloud*** (*VPC*) hace la función de red, donde podremos realizar todo nuestro despliegue, lanzando instancias, subredes y todo tipo de elementos. A estas *VPCs* se les ha de asignar un rango de direcciones IP al ser creadas, siendo la máscara "/24" la más común. Por lo tanto, el formato de estas IPs será "IP/MÁSCARA".

A continuación, tenemos las **subredes** que se pueden crear dentro de las *VPCs*. Cuando creamos una subred se le ha de indicar a que red (*VPC*) pertenece, el rango de IPs de la subred y la zona de disponibilidad en la que se encuentra, concepto que se explicará más adelante.

AWS también cuenta con sus propias **tablas de rutas** que se les ha de asignar a cada subred. Aquí únicamente se les ha de indicar a que *VPC* pertenecen y, una vez creadas, se configuran todas las rutas deseadas.

En este servicio nos encontramos también con el ***Internet Gateway*** que, tal y como indica el nombre, es la puerta de acceso que tendremos hacia Internet desde cualquiera de nuestras *VPCs*. Ésta simplemente se crea indicándole el nombre y luego ya se asigna a su correspondiente *VPC*.

Para finalizar con este servicio de AWS, nos encontramos con los ***Endpoints*** y ***Endpoints Services***. Estos dos elementos son los que nos permitirán conectar varias *VPCs*, creando un servicio con el *Endpoint Service* y usando los *Endpoints* para inteconectarlos.

3.2. Elastic Compute Cloud

El siguiente servicio que usamos es el *EC2*, que se encarga de las instancias de AWS, sus configuraciones, su correcto funcionamiento, los distintos tipos de balanceadores de carga y la creación de una pareja de claves.

El primer elemento que encontramos en este servicio son las **instancias**, donde encontramos una gran cantidad de sistemas operativos y versiones para usar. Para poder lanzar una instancia correctamente se han de realizar varias configuraciones, por lo que veremos las más importantes por encima y más adelante ya entraremos en detalle.

Lo primero que tenemos que escoger al lanzar una instancia es el sistema operativo y la versión que queremos. Una vez escogido esto, nos encontramos con los distintos tipos de instancia dentro de cada sistema operativo. Estos tipos de instancia tienen distintas combinaciones de CPU, memoria, almacenamiento y capacidad de red, adaptándose cada tipo a un caso de uso distinto.

Seguidamente, está la configuración propia de la instancia, donde se seleccionará la *VPC* en la que queremos que se encuentre, la subred donde se va a lanzar y el número de instancias a lanzar idénticas, entre otras configuraciones.

A continuación, nos encontramos con la adición de almacenamiento, donde podemos personalizar el tipo de volumen y el tamaño de este. También se le puede añadir distintas etiquetas a la instancia para identificarla.

Y, por último, se le ha de asignar un *Security Group*, un elemento de este mismo servicio que explicaremos a continuación. Aquí tenemos la posibilidad de crear uno nuevo para esta instancia o utilizar alguno que ya esté creado.

Una vez visto por encima como se crea una instancia, llegamos al punto de la seguridad para esta, un elemento llamado ***Security Group***. Los *Security Group* hacen la función de cortafuegos, ya que dentro encontramos las reglas de entrada y salida de datos para cada instancia.

Los **balanceadores de carga** también se encuentran en este servicio y, cuando nos disponemos a crearlos, nos encontramos con que hay distintos tipos, distinguiéndose por la capa en la que se encuentran. Estos tipos de balanceadores de carga son:

- ***Application Load Balancer (ALB):*** este primer balanceador de carga se encuentra en la séptima capa del modelo OSI, es decir, se encuentra a nivel de aplicación.

La cantidad de solicitudes que puede redireccionar este balanceador se encuentra limitado ya que ha de analizar cada solicitud, evaluando las reglas establecidas.

- ***Network Load Balancer (NLB):*** el balanceador de carga de red está en capas más bajas que el anterior, este se encuentra en la cuarta capa del modelo OSI, en el nivel de transporte. Este balanceador de carga, a diferencia del *Application Load Balancer*, puede atender a millones de solicitudes por segundo, ya que solo intenta abrir conexiones TCP con el destino de cada solicitud.
- ***Gateway Load Balancer (GWLB):*** este balanceador de carga está situado en la tercera capa del modelo OSI, a nivel de red. Es usado para aumentar la seguridad del sistema, ya que se suele usar como cortafuegos, sistema de prevención y detección de intrusiones y sistema de inspección de paquetes.
- ***Classic Load Blancer (CLB):*** finalmente, el balanceador de carga clásico es un balanceador en desuso y a punto de desaparecer, ya que no ofrece ninguna funcionalidad nueva. Lo único que hace es redistribuir la información según las reglas que se le configuren.

Para que estos balanceadores de carga sean funcionales, se ha de especificar las instancias con las que van a trabajar en un elemento llamado ***Target Group***. Cuando se crea un grupo de destino se puede especificar qué tipo de objetivo tendrá, es decir, si los elementos a los que va a redireccionar serán direcciones IP, instancias directamente o funciones lambda. También se decide el protocolo que se va a usar y en que *VPC* se encontrará.

Finalmente, está la creación de una **pareja de claves**, clave publica y clave privada, que nos va a permitir el acceso a las instancias que vayamos a crear.

4. Despliegue de un sistema seguro

Una vez hecha la introducción sobre el proyecto, el entorno y los elementos a usar, procederemos a la preparación del entorno de trabajo, a la propuesta del diseño y, finalmente su implementación.

4.1. Requisitos y preparación del entorno

Antes de empezar, tenemos que crearnos una cuenta de AWS que, como ya hemos mencionado anteriormente, es de pago con 12 meses gratis de prueba. Aun siendo este primer año gratuito, es necesario disponer de una tarjeta pago. También será obligatorio disponer de un número de teléfono con el que se verificará nuestra identidad.

Para poder crearnos un usuario en la plataforma tenemos que acceder a la siguiente URL: https://aws.amazon.com/es/

Una vez dentro, en la esquina superior derecha encontramos el botón de iniciar sesión donde, dentro, tenemos la opción de crear una cuenta nueva. La primera pantalla que nos encontramos es para especificar el correo y la contraseña de nuestra cuenta.

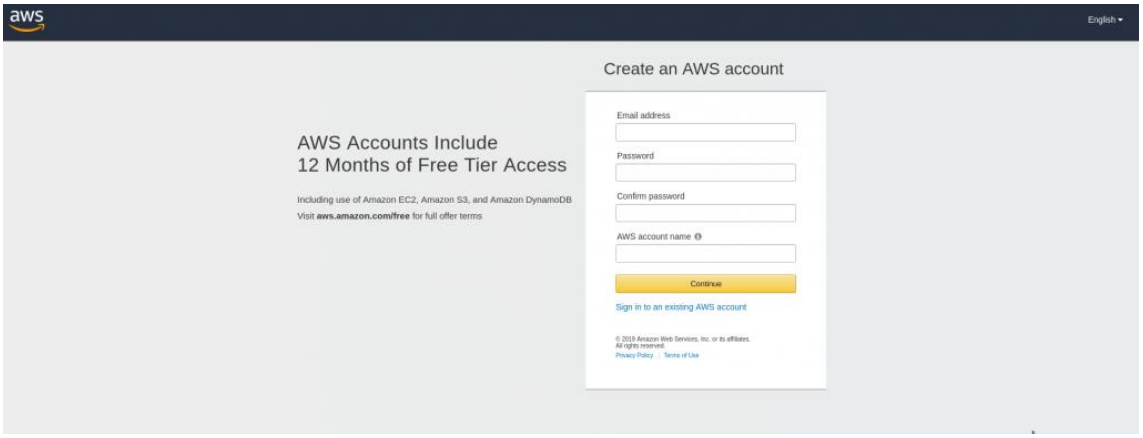

Figura 4.1 Datos del usuario

Una vez tengamos el correo, la contraseña y el nombre, entramos a la pantalla donde tenemos todos los datos personales del usuario. Aquí encontramos un punto importante y es en el momento de escoger el tipo de cuenta, ya que se puede crear tanto una cuenta para una empresa como una cuenta personal. En nuestro caso escogemos una cuenta personal.

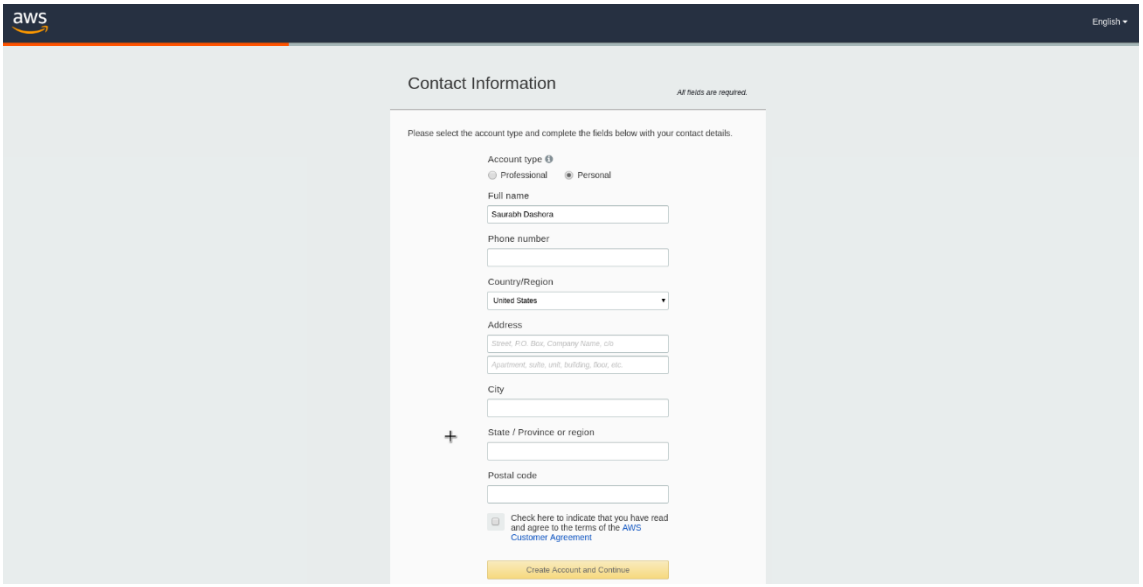

Figura 4.2 Datos personales del usuario

El tercer paso con el que nos encontramos es para la información de nuestra tarjeta bancaria que usaremos para pagar mes a mes. Amazon nos hará una pequeña verificación reteniendo temporalmente 1€ para asegurarse de la validez de la cuenta.

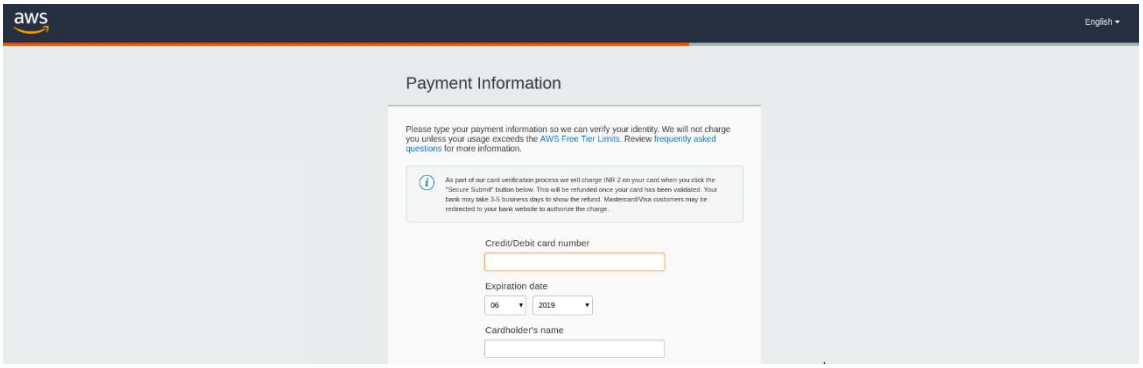

Figura 4.3 Datos bancarios

Una vez rellenados estos datos, nos encontramos con una pantalla de verificación de identidad, donde tendremos que introducir nuestro número de teléfono en el que recibiremos un SMS con un código.

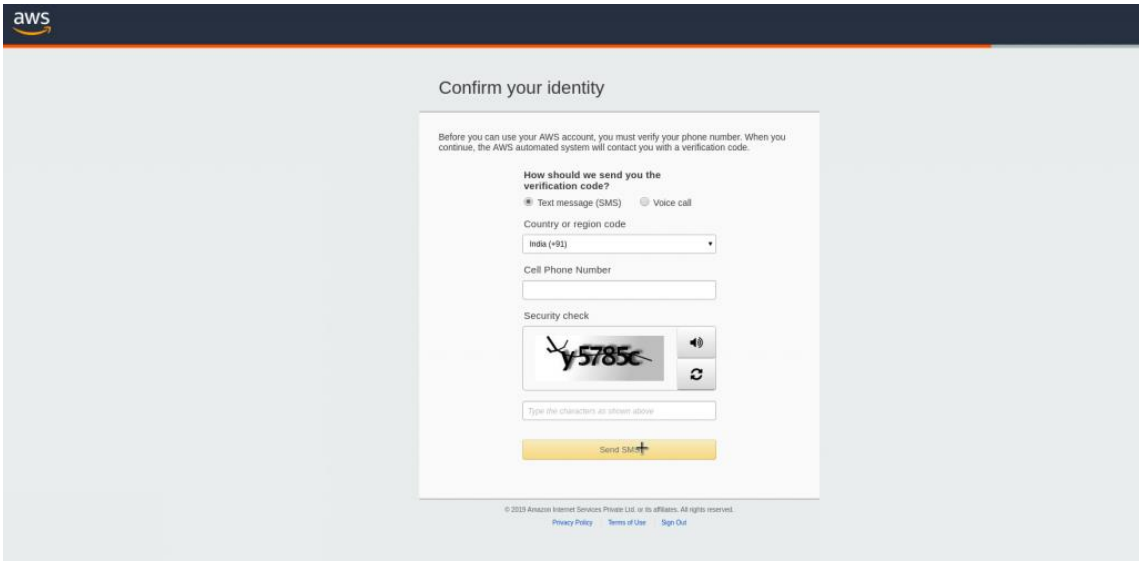

Figura 4.4 Confirmación de identidad

Finalmente, una vez verificada nuestra identidad, tenemos que seleccionar el tipo de suscripción que queramos. En este punto, escogemos el plan básico, ya que es el gratuito y es suficiente para realizar todo lo que necesitamos.

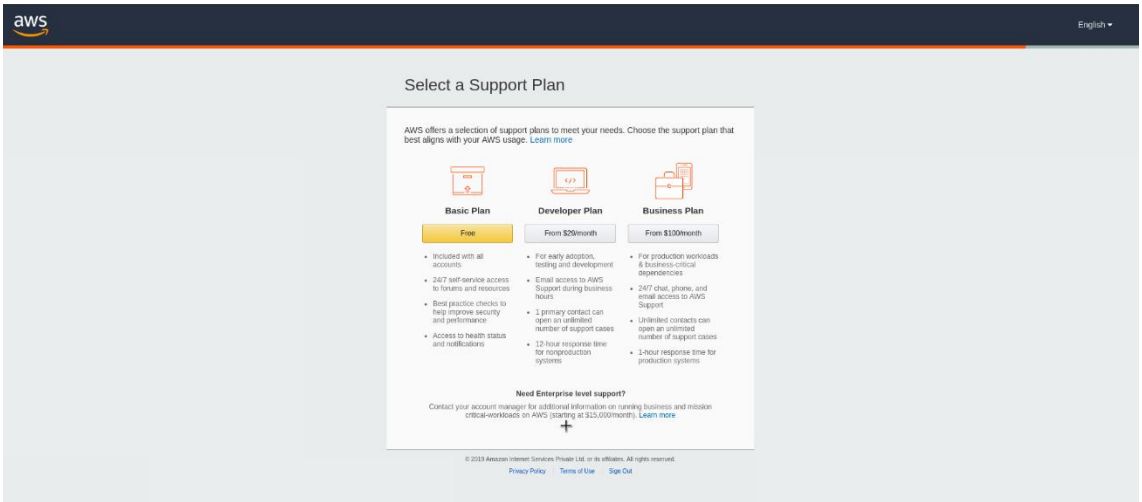

Figura 4.5 Tipo de suscripción

Una vez creada la cuenta, empezamos con el *Free Tier*, donde dispondremos de una gran cantidad de funcionalidades gratuitas durante 12 meses.

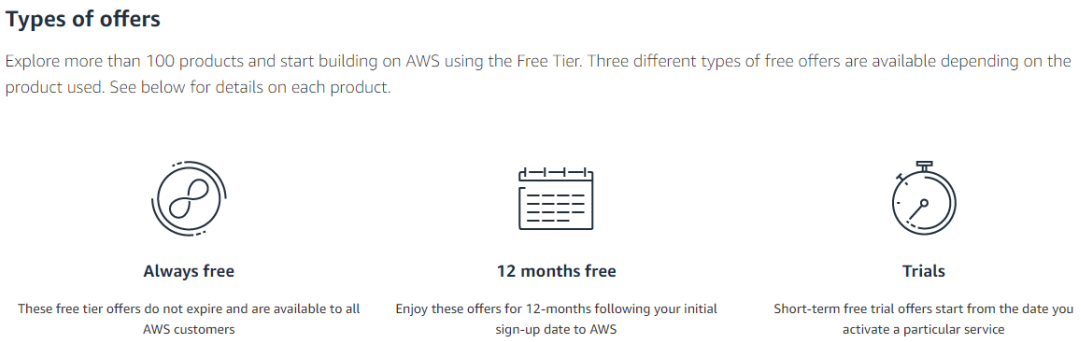

Figura 4.6 Free Tier

Una vez creada la cuenta y habiendo iniciado sesión, tenemos que escoger en que región queremos trabajar. Cada región tiene su número de zonas de disponibilidad donde podremos trabajar. Una zona de disponibilidad es un *Cluster* de *DataCenters*, característica que permite a AWS tener una alta resiliencia. Estas zonas están conectadas entre sí con una baja latencia para permitir pequeñas conexiones pero lo suficientemente separadas para ser más tolerantes a fallos. En nuestro caso, nos encontramos en la zona de Ohio, con tres zonas de disponibilidad.

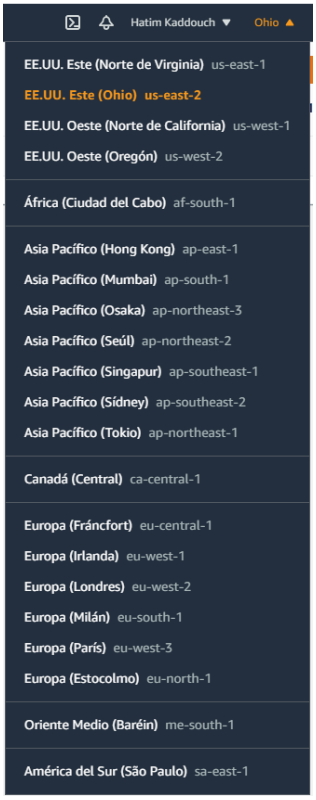

Figura 4.7 Regiones y zonas de disponibilidad actualmente

Una vez seleccionada nuestra zona, procedemos a crear nuestra pareja de claves, a que serán necesarias para cuando tengamos que acceder a una instancia. Para ello, accedemos a la pestaña de servicios y escogemos el servicio *EC2*. En el apartado de "Red y Seguridad" encontramos el botón de "Pares de claves". Una vez le damos a crear par de claves nos encontramos con la siguiente pestaña.

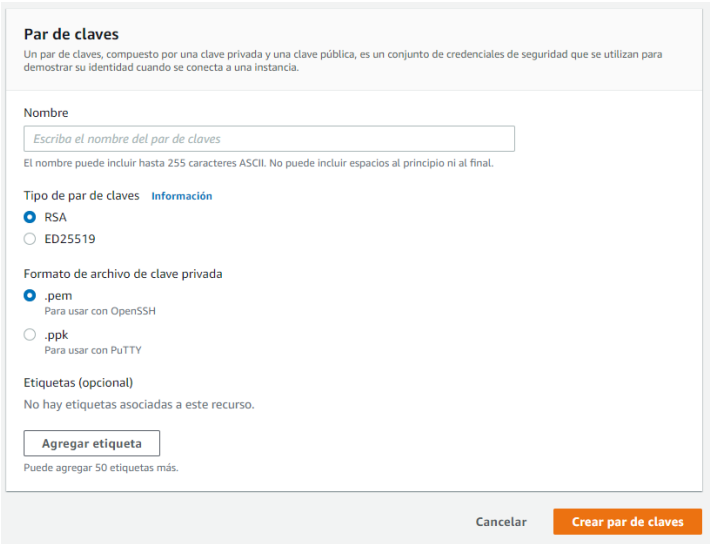

Figura 4.8 Creación de par de claves

En nuestro caso, ciframos con RSA y escogemos el formato .pem ya que luego podremos realizar una transformación mediante PuTTYgen para acceder a nuestras instancias. Una vez tengamos descargado este fichero, tendremos que descargar PuTTY y, junto a él, se nos descargará PuTTYgen, la herramienta para generar claves privadas para PuTTY.

Una vez instalado, encendemos PuTTYgen e insertamos nuestra clave privada que nos hemos descargado. A continuación, configuramos los parámetros necesarios.

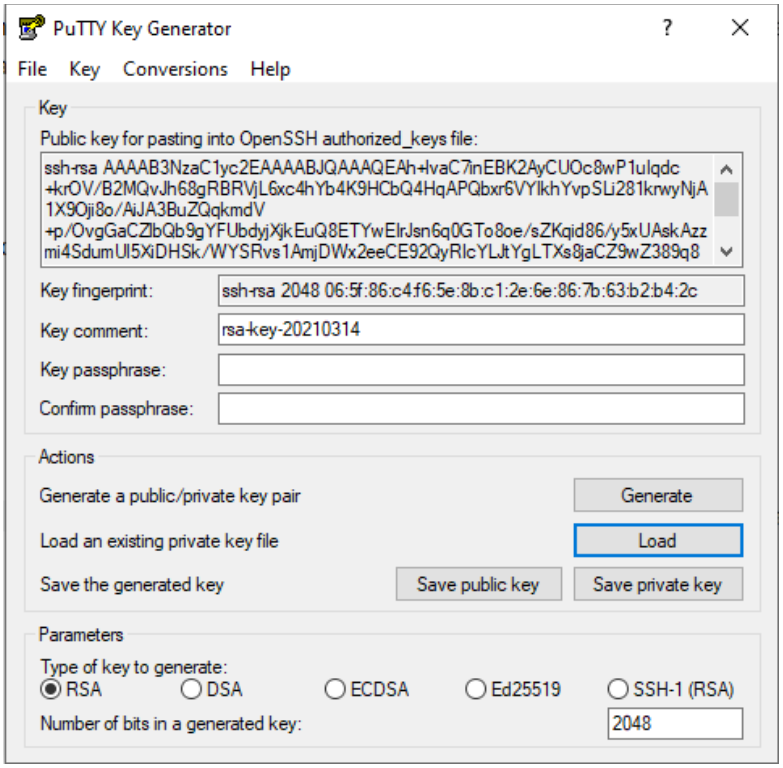

Figura 4.9 Transformación clave privada para PuTTY

Una vez hecho todo esto, ya podremos empezar a trabajar en nuestra cuenta de AWS.

4.2. Propuesta y diseño

Una primera idea para realizar este sistema fue crear una única *VPC* en la que se haría todo el montaje mediante subredes y los diferentes balanceadores de carga que usaremos. Esta idea era buena ya que dentro de una *VPC* podemos crear un gran número de subredes, pero analizándola mejor vi que perdía potencial. El principal problema fue la seguridad, ya que conectando subredes entre ellas resulta ser menos seguro que haciendo el montaje en diferentes *VPCs*. El añadir más *VPCs*, aparte de añadirle seguridad, también le añade dificultad al diseño.

Viendo esto, nos planteamos hacer el sistema en tres *VPCs* distintas que tendrán diferentes funcionalidades. Las dos primeras *VPCs* que nos encontramos contienen los

servidores que vamos a lanzar. Estas dos *VPCs* tienen la misma estructura, por lo que todo lo que expliquemos de una se puede extrapolar a la otra.

En esta *VPC* encontramos dos subredes distintas, cada una en una zona de disponibilidad distinta, con el mismo servidor dentro. De esta manera, con esta redundancia en diferentes zonas de disponibilidad, conseguimos una alta tolerancia al fallo, ya que, si falla uno de los servidores, tenemos disponible el otro.

Estas dos *VPCs* también tenemos un *Network Load Balancer* en cada una que nos redirigirá los datos a un servidor u otro.

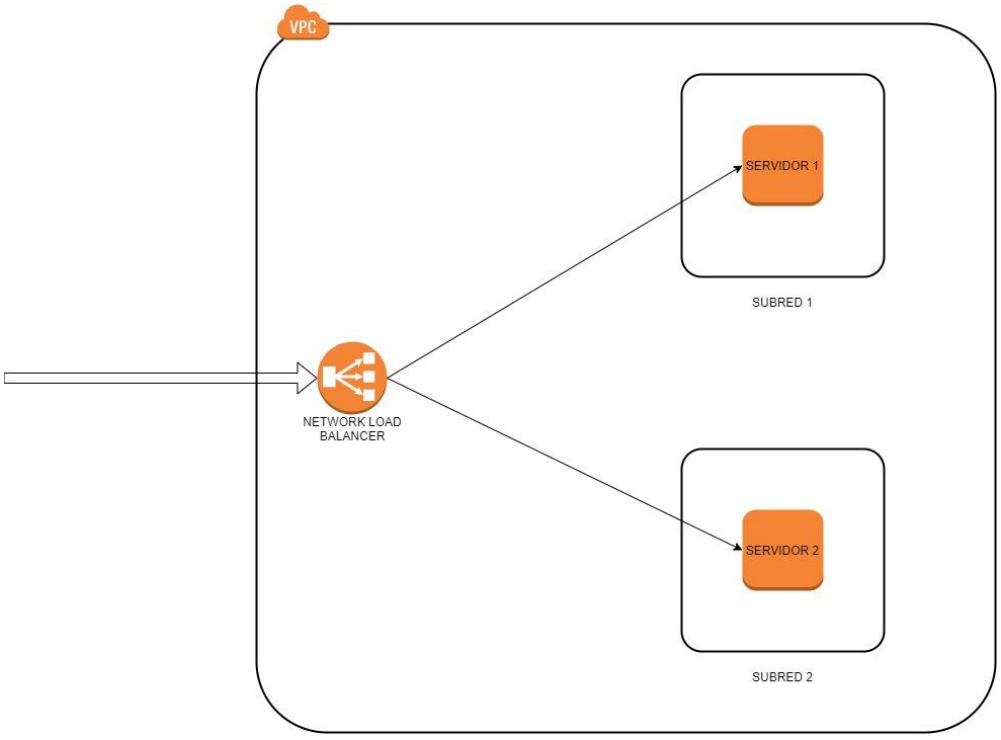

Figura 4.10 Diagrama VPCs de los servidores

La otra *VPC* que nos encontramos hace el papel de DMZ. En esta *VPC* encontramos dos subredes distintas que contienen nuestros *Endpoints* que nos servirán para conectarnos a las otras dos *VPCs* ya mencionadas. Cada subred tiene dos *Endpoints*, uno para cada *VPC* con servidor. Estos *Endpoints* se conectarán a las otras *VPCs* mediante un servicio llamado *PrivateLink*, un servicio que permite conectar *VPCs*, otros servicios de AWS y otros elementos sin exponer el tráfico a la Internet pública. Este punto le añade más seguridad a nuestro sistema, ya que nos permite mantener nuestras dos *VPCs* con los servidores completamente aislada de Internet. El único punto de acceso a estos servidores es mediante la DMZ.

Finalmente, en esta *VPC* también tendremos un *Application Load Balancer* que nos distribuirá las peticiones entrantes entre nuestras dos subredes y, de aquí, se irá al servicio que se esté pidiendo.

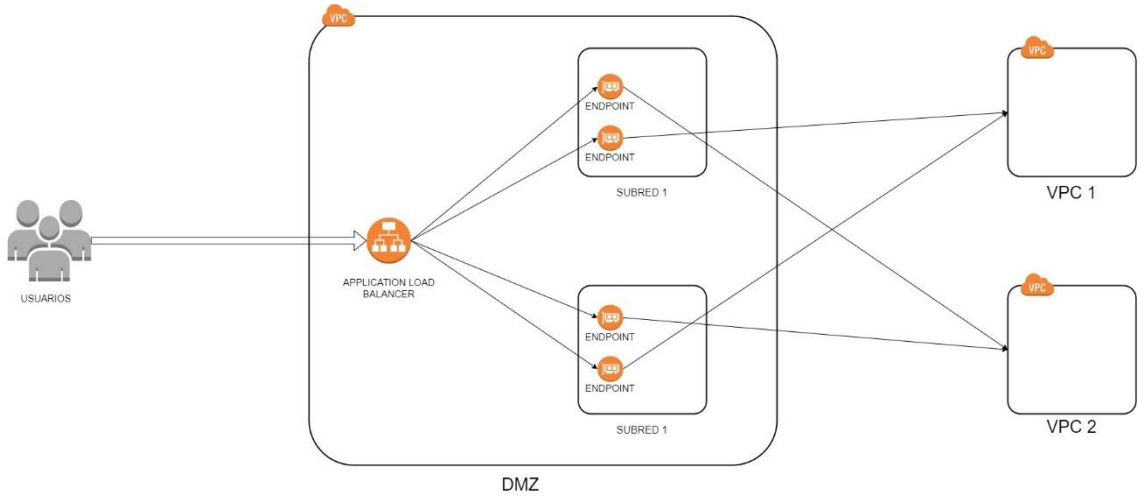

Figura 4.11 Diagrama de la DMZ

Finalmente, el montaje entero quedaría de la siguiente manera, conectando los *Endpoints* a los *Network Load Balancers* de las otras dos *VPCs*.

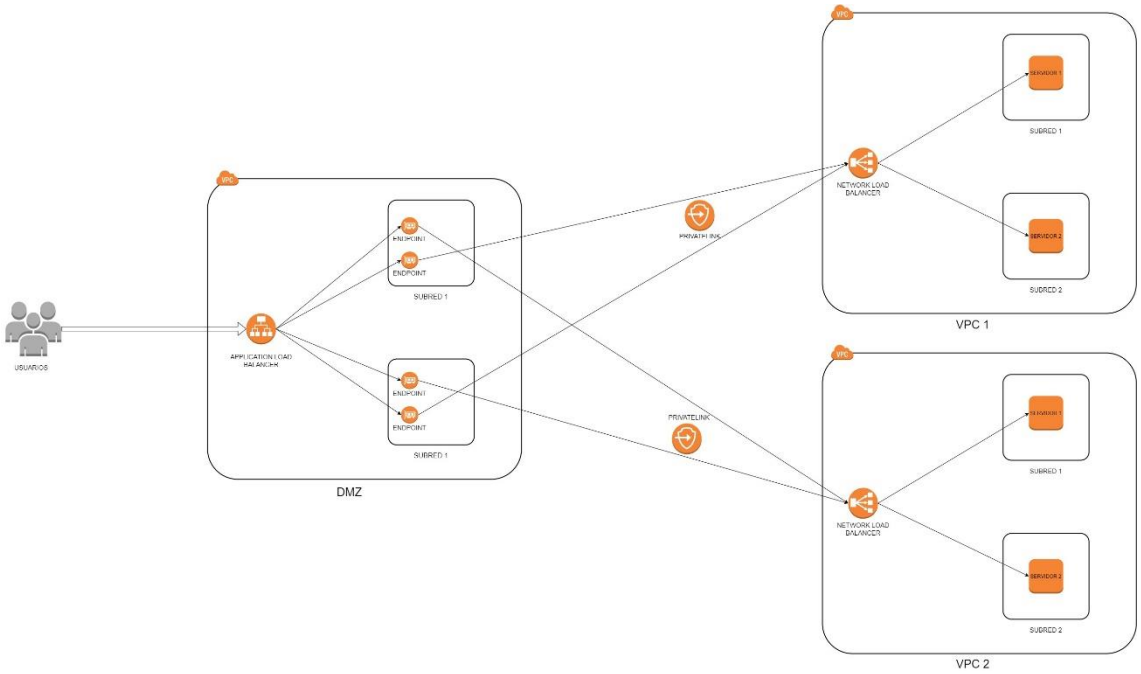

Figura 4.12 Diagrama final

4.3. Implementación

Para realizar una correcta implementación de este sistema, se empezará montando las *VPCs* que contienen los servidores y después la DMZ.

Lo primero que tenemos que hacer es crear nuestras *VPCs*, en este caso, tres. Aunque de momento no trabajemos en la DMZ, iremos creando los contenedores para avanzar trabajo.

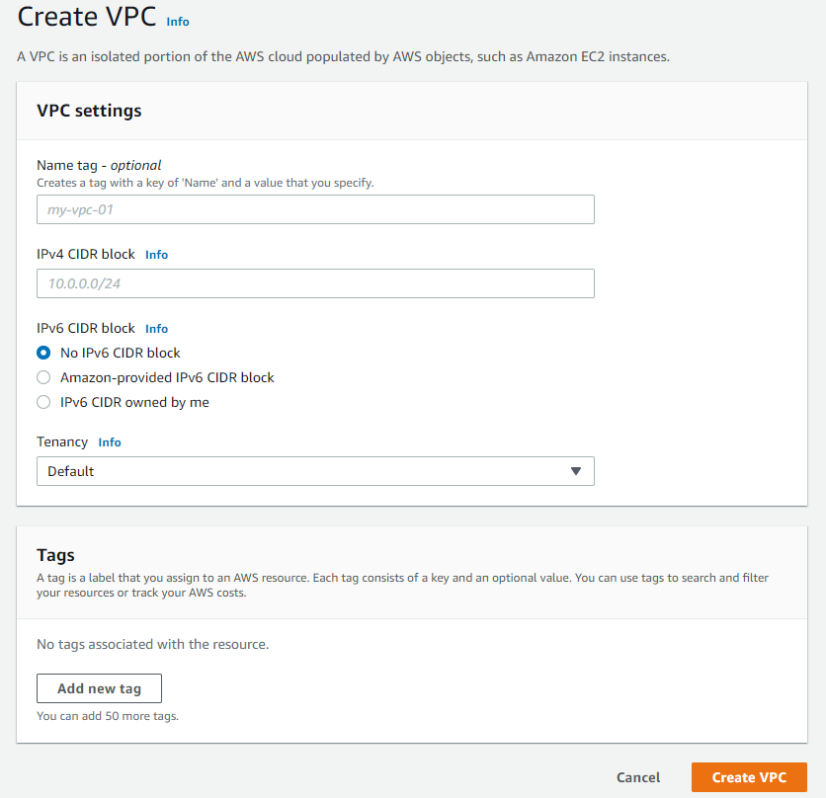

Figura 4.13 Creación de las VPCs

Como podemos observar, a la hora de crear las *VPCs* nos piden el nombre de esta y su rango de direcciones. Las tres que vamos a crear tendrán los siguientes parámetros:

- ***VPC* 1:** esta primera *VPC* contendrá nuestro primer servidor que se le asignará el color azul para poder diferenciarlo del otro. Como nombre tendrá "*Blue VPC*" y un rango de IPs 10.100.0.0/16.
- ***VPC* 2:** nuestra segunda *VPC* tendrá el segundo servidor que se le asignará el color verde. Como nombre tendrá "*Green VPC*" y un rango de IPs 10.200.0.0/16.
- ***VPC* 3:** finalmente, la tercera *VPC* será nuestra DMZ, por lo que tendrá de nombre "DMZ" y un rango de IPs 10.0.0.0/16.

Escogemos la máscara /16 ya que nos permite tener más cantidad de subredes a cambio de reducir el número de instancias en cada una. Esto nos va a beneficiar en caso de que queramos expandir el sistema, ya que preferimos tener más subredes para tener una mejor tolerancia al fallo.

Una vez creadas las *VPCs*, se ha de modificar una pequeña característica para permitir que las instancias en estas *VPCs* puedan obtener sus nombres de host DNS. Para ello, pulsamos con el botón derecho sobre la *VPC* y seleccionamos la opción de *"Enable DNS hostnames"*.

Una vez tenemos las *VPCs*, tenemos que crear y preparar todas nuestras subredes. Para ello, en este mismo panel, encontramos la pestaña de subredes.

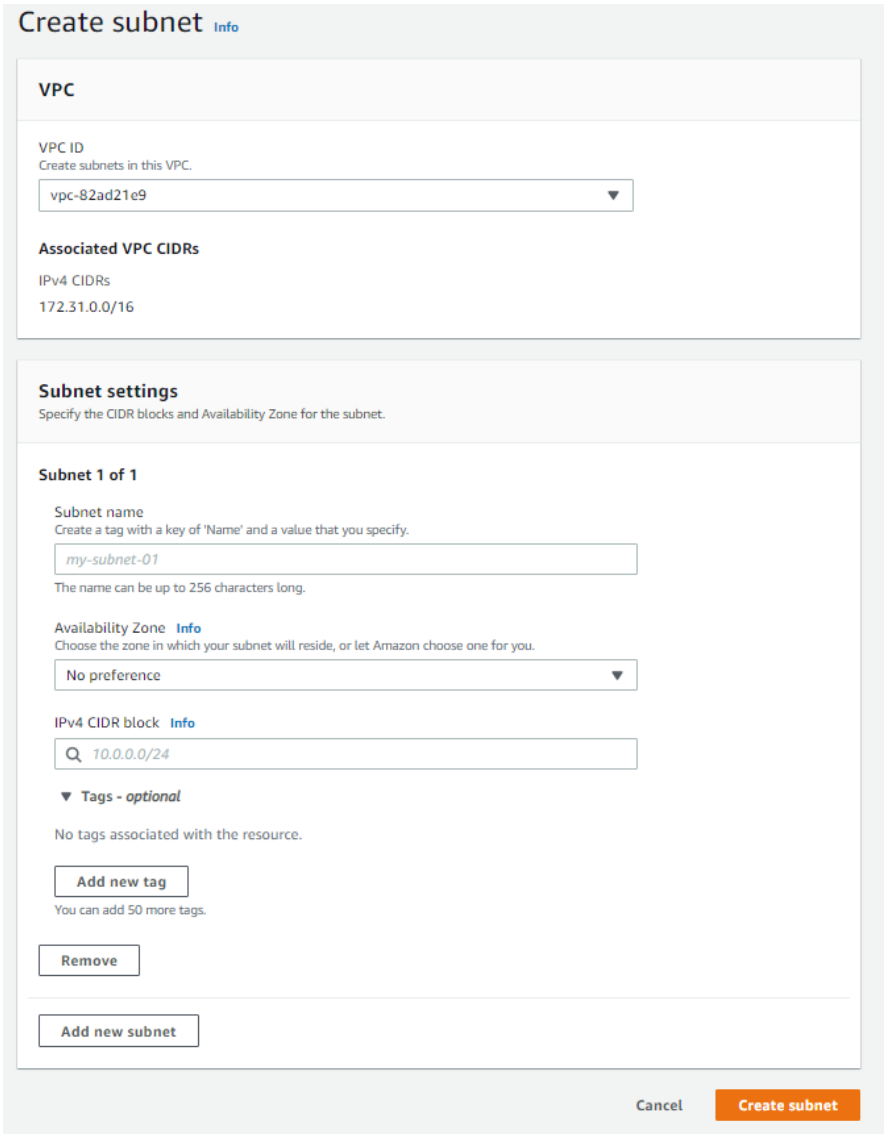

Figura 4.14 Creación de las subredes

Para cada *VPC* que tenemos creada, le crearemos dos subredes como hemos visto anteriormente en el diagrama del montaje. En este punto, tenemos que seleccionar la *VPC* donde queremos tener nuestra subred, el nombre que se le asignará, la zona de disponibilidad que va a diferir entre las dos subredes dentro de cada *VPC* y finalmente, el rango de IPs. Para estas configuraciones de cada subred tendremos los siguientes parámetros:

- **Blue VPC:** el nombre que se les asigna es *"Blue Subnet 1"* y *"Blue Subnet 2"*, encontrándose en las zonas de disponibilidad "us-east-2a" y "us-east-2b" respectivamente. Para el rango de IPs tendremos 10.100.1.0/24 y 10.100.2.0/24.
- **Green VPC:** para esta *VPC* haremos lo mismo, llamando las subredes *"Green Subnet 1"* y *"Green Subnet 2"*, en las zonas de disponibilidad "us-east-2a" y "us-east-2b" respectivamente y con los rangos de IP 10.200.1.0/24 y 10.200.2.0/24.
- **DMZ:** finalmente, para la DMZ tenemos *"DMZ Subnet 1"* y *"DMZ Subnet 2"* en las zonas de disponibilidad "us-east-2a" y "us-east-2b", con sus rangos de 10.0.1.0/24 y 10.0.2.0/24.

En este caso también tenemos que cambiar una opción, y es que cuando creamos una subred nueva, esta no asigna IPs públicas a las instancias que se lancen en ella. Para ello, con el botón derecho del ratón, cambiamos la opción de *"Modify auto-assign IP settings"*.

Nuestro siguiente paso es proporcionarles acceso a internet a nuestras *VPCs*, creando el *Internet Gateway* en esta misma pantalla. Para ello le damos clic al botón de *Internet Gateways* y lo creamos. En este caso, vamos a crear tres puertos de entrada para Internet, ya que la DMZ necesita acceso a Internet y en las redes de los servidores lo necesitaremos para poder acceder a las instancias para configurarlas. Más adelante se eliminarán los dos puertos de las redes de los servidores para que únicamente se pueda acceder mediante la DMZ. Una vez creados los *Gateways*, se adjuntan a sus respectivas *VPCs* con el clic derecho y en la opción de adjuntar.

El último paso que vamos a hacer para terminar de configurar nuestras *VPCs* y subredes será crear sus tablas de rutas, una para cada *VPCs*, que nos servirán para tener acceso a internet.

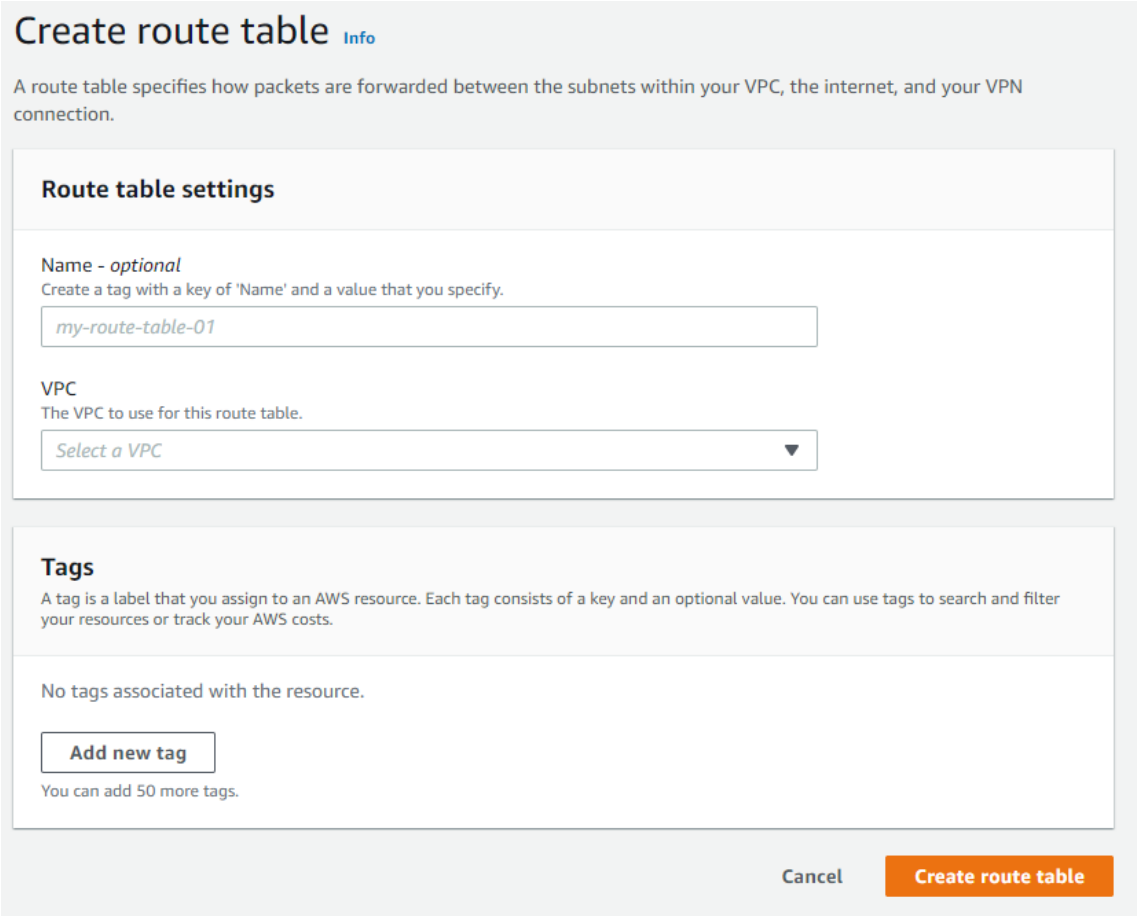

Figura 4.15 Creación de la tabla de rutas

Para la creación de estas tablas de rutas solo se tiene que especificar el nombre de la tabla y la *VPC* a la que pertenece. Una vez creada, es cuando se han de configurar las rutas, yendo a la tabla creada y clicando en la opción de editar rutas.

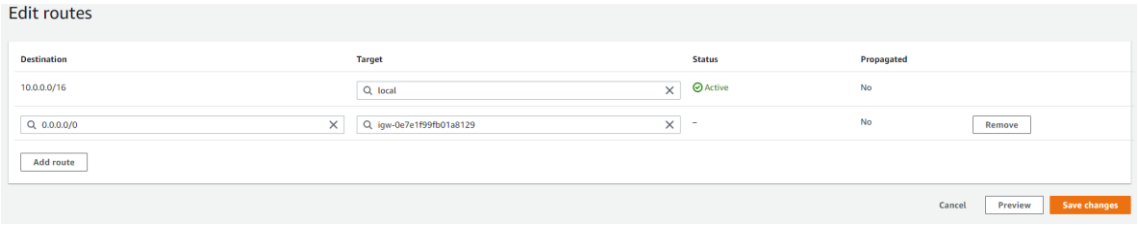

Figura 4.16 Edición de las rutas

Como podemos observar, hay una ruta predeterminada que no se puede modificar, que es la que nos indica que toda la información que se dirija a la IP de nuestra *VPC* tenga como objetivo "local", refiriéndose a la misma *VPC*. La segunda regla es la que tenemos que añadir nosotros, que es la que nos permitirá tener acceso a Internet. En esta regla marcamos como objetivo nuestro *Internet Gateway* creado anteriormente y como origen la IP 0.0.0.0/0, la IP con la que se identifica Internet. Como último paso, tenemos que asociar esta tabla de enrutamiento a sus correspondientes subredes.

Una vez tenemos las *VPCs* con sus correspondientes subredes configuradas, cambiamos de servicio al *EC2*, donde empezaremos a crear nuestras instancias que harán de servidores. Lo primero que preparamos son los *Security Groups*.

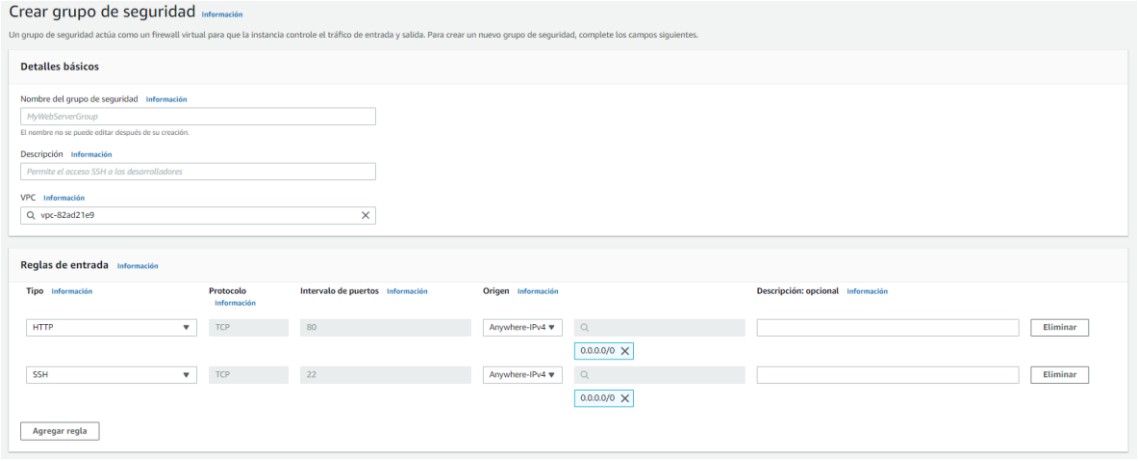

Figura 4.17 Creación de los Security Group

Para los *Security Groups*, seleccionamos el nombre que se le asignará junto a una pequeña descripción y a que *VPC* pertenece. Crearemos un *Security Group* para cada *VPC*.

En las reglas de entrada, daremos permiso a los protocolos SSH y HTTP para poder acceder a nuestras instancias y tener acceso a internet. Para las reglas de salida dejaremos las de por defecto que permite la salida de todo.

Hecho esto, llegamos al punto donde se lanzan las distintas instancias. Una vez le damos a lanzar instancia, podemos observar que tenemos una gran variedad de sistemas operativos. En este caso, usaremos instancias Ubuntu, concretamente, Ubuntu Server 20.04 LTS. Usamos esta instancia ya que, por una parte, es un sistema operativo con el que estamos familiarizados y, por otra, lo tenemos de forma gratuita.

El segundo paso es escoger el tipo de instancia que queramos, donde dejaremos la que tenemos por defecto ya que es suficiente para lo que buscamos hacer y es la única gratuita.

En el siguiente paso encontramos la configuración de la instancia, donde veremos una gran cantidad de configuraciones y características. Nosotros nos centraremos únicamente en el tercer bloque de configuraciones.

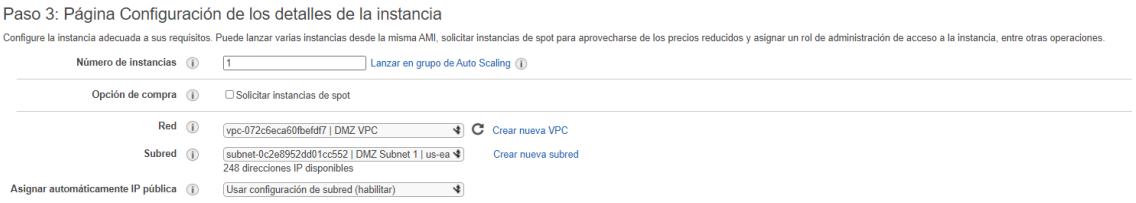

Figura 4.18 Configuración de la instancia

En este bloque se nos pide seleccionar una *VPC* y una subred dentro de esta. La tercera característica que tenemos es la asignación de IPs públicas, la cual pondremos la opción de "Usar configuración de subred (habilitar)". Esta opción estará deshabilitada en el caso de que no se haya configurado debidamente nuestras subredes.

A continuación, tenemos la configuración de almacenamiento, la cual dejaremos la que encontramos por defecto, y la agregación de etiquetas, donde podemos añadir todas las que queramos.

Finalmente, entramos en el apartado de *Security Groups*, en el cual seleccionaremos el creado en pasos anteriores.

Una vez le demos a lanzar, tenemos que seleccionar nuestro par de claves generado en la preparación del entorno.

Hacemos este mismo proceso para las cuatro instancias que vamos a lanzar en nuestras subredes *"Blue Subnet 1"*, *"Blue Subnet 2"*, *"Green Subnet 1"* y *"Green Subnet 2"*.

Una vez tengamos nuestras instancias en ejecución y con todas las comprobaciones automáticas hechas, tenemos que acceder a ellas para configurar nuestro servidor. Para facilitar esta parte, haremos una instalación y configuración de un servidor HTTP Apache.

Para acceder a ellas y tener nuestra línea de comandos para la configuración, usaremos PuTTY.

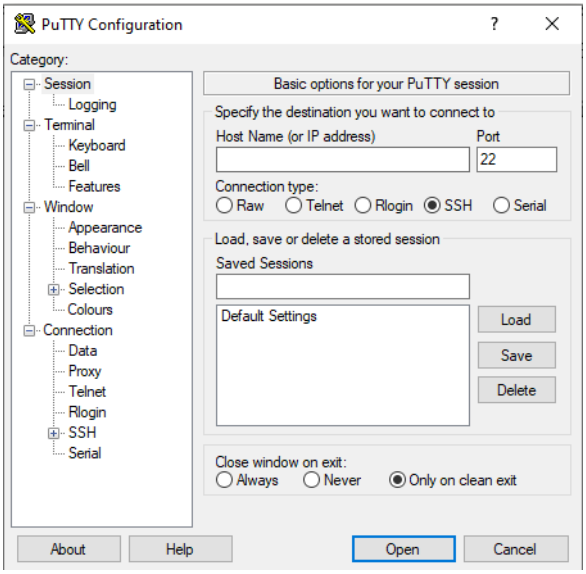

Figura 4.19 Entrada a la instancia mediante PuTTY

En esta primera pantalla de PuTTY, tenemos que especificar el nombre del host al que queremos acceder. El formato de este nombre para AWS es el siguiente:

nombre-de-usuario-instancia@ip-publica-instancia

Para el sistema operativo que hemos escogido nosotros, el nombre de usuario es Ubuntu y, la IP pública de la instancia la encontramos en los detalles de esta. El tipo de conexión lo dejamos en SSH.

Una vez especificado el nombre del host, procederemos a introducir nuestra clave de acceso. Para ello, vamos la categoría SSH y, una vez dentro, a la subcategoria llamada AUTH.

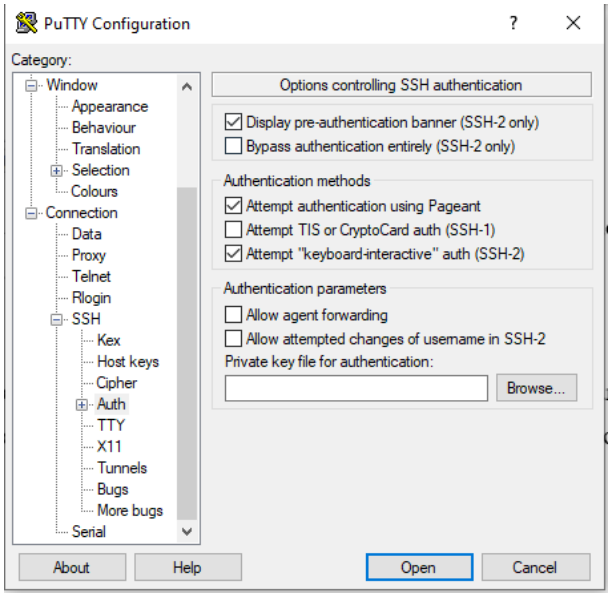

Figura 4.20 Entrada a la instancia mediante PuTTY

Aquí escogemos nuestra clave privada y ya podremos conectarnos a nuestra instancia.

Una vez dentro, lo primero es ponernos en modo *root* mediante la comanda *sudo su*. Estando ya en modo *root*, actualizaremos nuestros repositorios de paquetes e instalaremos Apache2 con las siguientes comandas:

apt-get update -y

apt-get install apache2 -y

Hecho esto, tendremos instalado nuestro servidor web y solo hará falta configurarlo. Una manera fácil y rápida para configurarlo es ejecutando una comanda similar a la siguiente:

*bash -c 'echo "

Blue Instance 1" > /var/www/html/index.html'*

Con esta comanda de ejemplo lo que estamos haciendo es guardar en el fichero */var/www/html/index.html* la secuencia escrita por el *echo*. Con esto estamos haciendo que el servidor, en vez de mostrarnos la pantalla de inicio de Apache2, nos devuelva el nombre de la instancia en su color, en este caso, azul. Una vez ejecutada esta comanda, volveremos a usar la misma redirigiendo la salida del *echo* a otro fichero llamado *green.html* o *blue.html*, dependiendo de la máquina en la que nos encontremos. Esto será importante ya que mediante el *Application Load Balancer* que crearemos más adelante podremos redirigir las peticiones al servidor azul o verde según queramos.

A continuación, accedemos al fichero */etc/spache2/apache2.conf* donde modificaremos el *AllowOverride* del directorio */var/ww/* de "*None"* a "*All"*.

```
<Directory /var/www/>
        Options Indexes FollowSymLinks
        AllowOverride All
        Require all granted
</Directory>
```

Figura 4.21 AllowOverride del Apache2

Para terminar, reiniciamos el servidor de apache para aplicar los cambios con la comanda *"systemctl restart apache2.service"*. Este proceso se ha de realizar para las cuatro instancias que tenemos, asignándole el nombre debido a cada una dentro de los ficheros *index.html* y *blue.html/green.html*.

Una vez tengamos esto, podemos probar si funciona introduciendo en nuestro buscador la IP pública de la instancia.

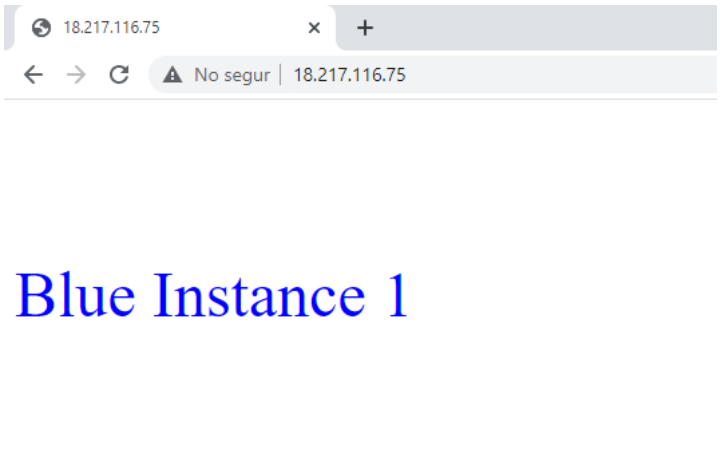

Figura 4.22 Prueba del servidor

Para terminar con las *VPCs* de los servidores, crearemos un *Network Load Balancer* en cada una de ellas el cual nos ayudara a redistribuir la carga de trabajo entre los dos servidores dentro de cada *VPC*.

El primer paso que hacer para el despliegue de nuestro balanceador de carga será crear los grupos de destino a los que vamos a enviar los paquetes. Esto se hace a través de la herramienta de grupos de destino, donde le daremos a crear grupo de destino.

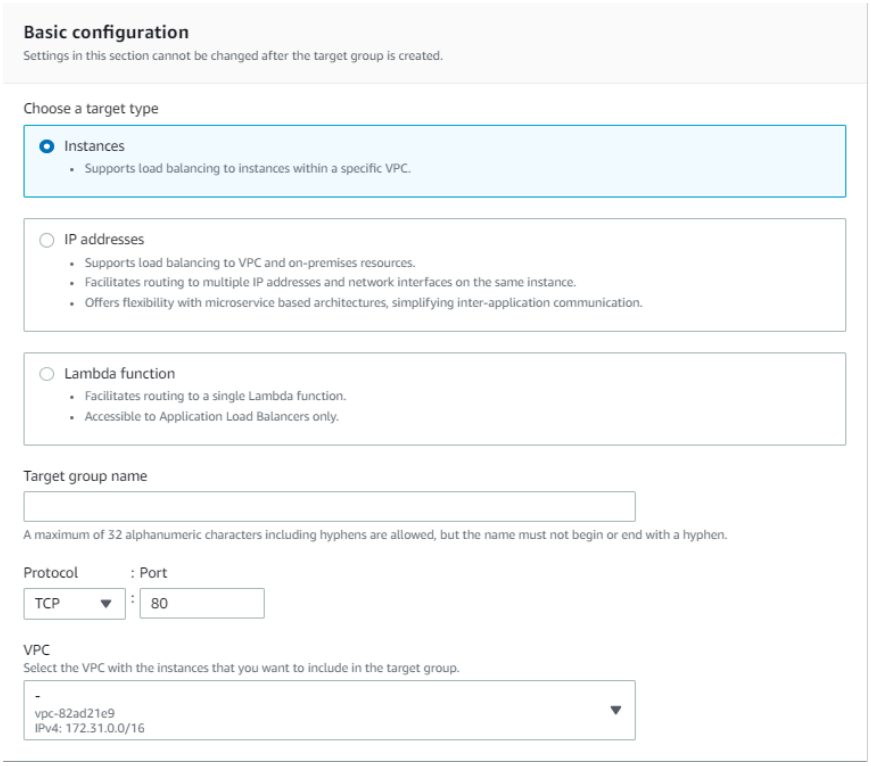

Figura 4.23 Creación del grupo de destino

En esta primera pantalla nos encontramos con la configuración básica del grupo de destino. Para el tipo de objetivo seleccionamos la opción de instancia, ya que este

balanceador de carga trabaja directamente con nuestros servidores. Para el protocolo y el puerto, dejaremos el protocolo TCP, ya que este balanceador de carga se encuentra directamente en la capa de transporte, y para el puerto, el número 80. En la *VPC* seleccionamos la correspondiente, ya que haremos dos distintas, una para el servicio azul y otro para el verde.

Una vez pasada esta pantalla, se nos pedirá registrar las instancias que queremos tener en nuestro grupo de destino.

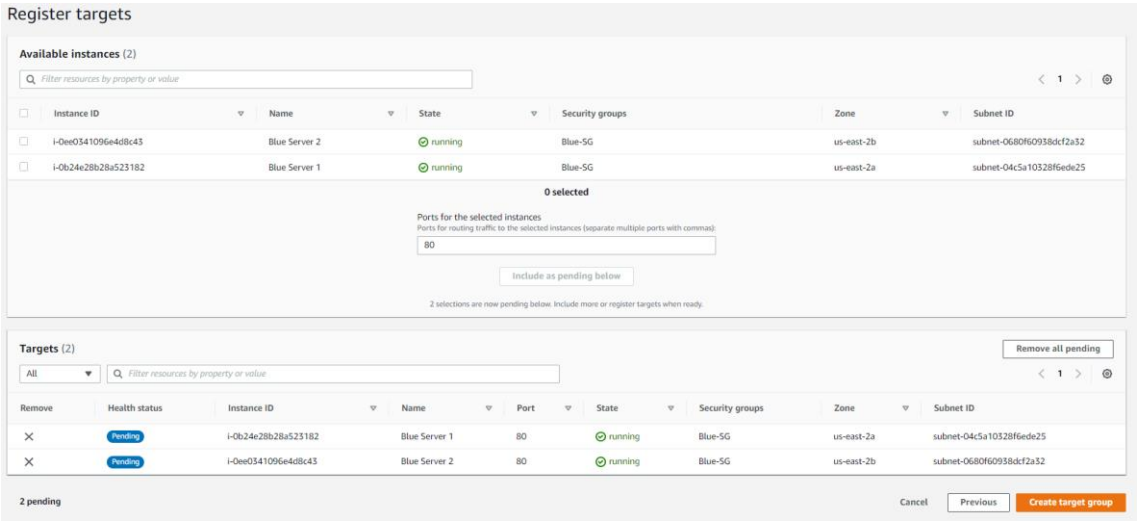

Figura 4.24 Registro de instancias al grupo de destino

Teniendo los dos grupos de destino preparados, comenzamos con la creación de los *Network Load Balancers*. Lo primero que nos aparece al crear el balanceador de carga es el tipo que queremos, en este caso, el *Network Load Balancer*. Para las configuraciones básicas, dejaremos los valores por defecto, introduciendo un nombre para el balanceador y seleccionando la *VPC* en la que se encuentra. Aquí el paso importante es en el momento de especificar nuestros agentes de escucha.

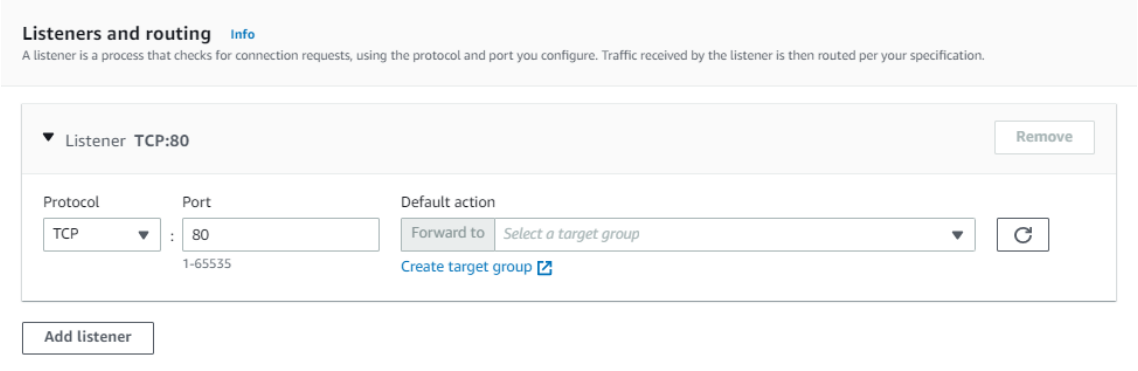

Figura 4.25 Agente de escucha para el balanceador de carga

Para el protocolo y el puerto dejamos los valores por defecto, TCP y puerto 80. Para la acción predeterminada, seleccionaremos el grupo de destino creado anteriormente, cada uno en su correspondiente *VPC*, es decir, el grupo de instancias azules en la *VPC* azul y el grupo verde en la *VPC* verde.

Una vez creado el balanceador de carga, es hora de comprobar que todo está correcto. Para ello, buscamos nuestro grupo destino y accedemos a él, donde veremos una pestaña llamada *Health Check*. Aquí podremos comprobar que todas las instancias están correctas.

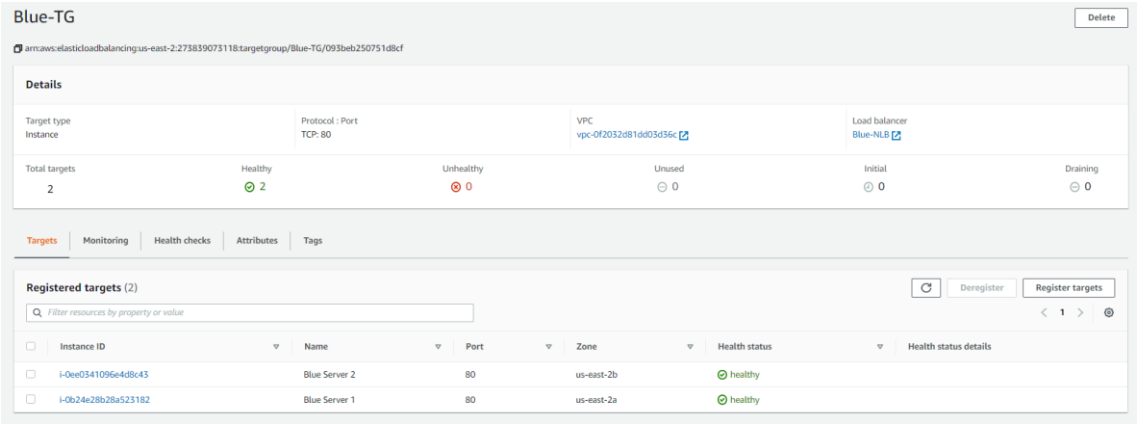

Figura 4.26 Health Check de las instancias

Aquí podemos comprobar que las dos instancias azules están en estado *Healthy*. Es posible que en un principio se encuentren en estado *Initial*, para que pase al estado verde solo hará falta esperar un rato.

Teniendo esto listo, podremos comprobar el funcionamiento del balanceador de carga introduciendo su DNS en un buscador web y veremos que irá alternando entre nuestras dos instancias.

Figura 4.27 Instancia 1 en el balanceador

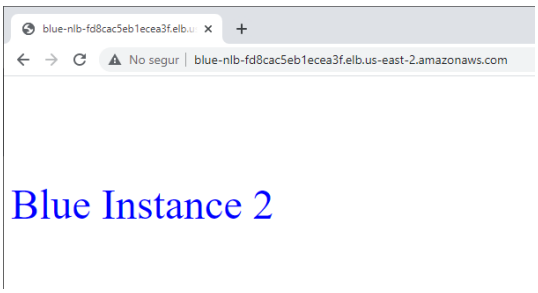

Figura 4.28 Instancia 2 en el balanceador

El siguiente paso es empezar con la DMZ y sus distintos elementos. Para ello, el primer paso que haremos es empezar a preparar el *PrivateLink* con el que conectaremos nuestros servidores a la DMZ.

Volvemos al servicio *VPC*, donde encontraremos los elementos *Endpoint* y *Endpoint Services*, que son los que vamos a usar para el *PrivateLink*. Entramos a *Endpoint Services* y empezamos creando el servicio para una de nuestras dos *VPCs* con servidores.

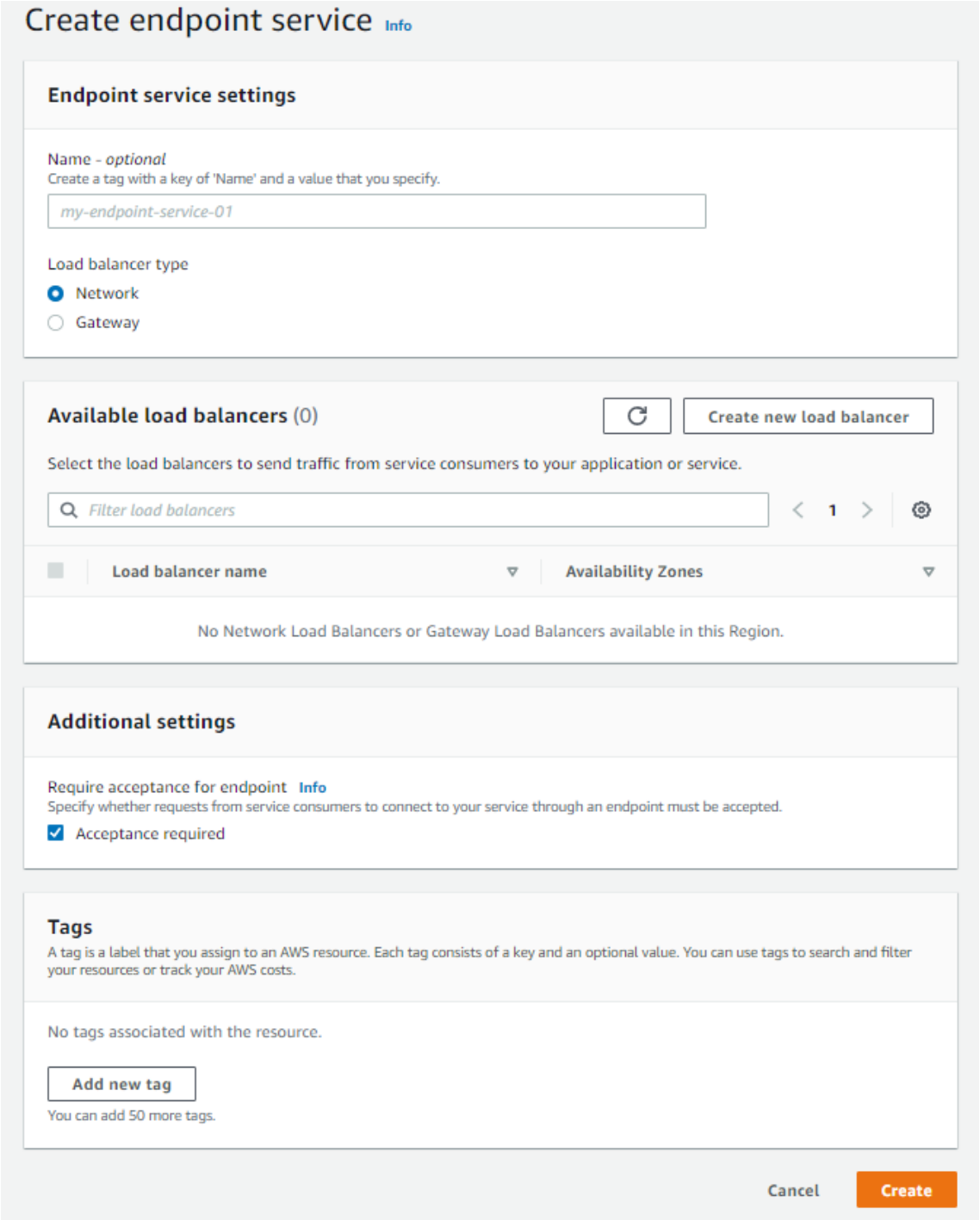

Figura 4.29 Creación del Endpoint Service

En esta pantalla vemos que nos piden especificar el nombre del servicio, el tipo de balanceador de carga al que vamos a el servicio, en nuestro caso, los dos eran de red y, finalmente, escoger el balanceador de carga que queramos. Dejamos la casilla de

"Acceptance requires" ya que cuando creemos los *Endpoints*, vamos a querer aceptarlos como punto de conexión.

Antes de continuar con la creación de estos *Endpoints*, copiaremos el nombre del servicio que acabamos de crear. Este lo encontraremos en su pestaña de detalles.

Teniendo este nombre copiado, vamos la creación de *Endpoints*.

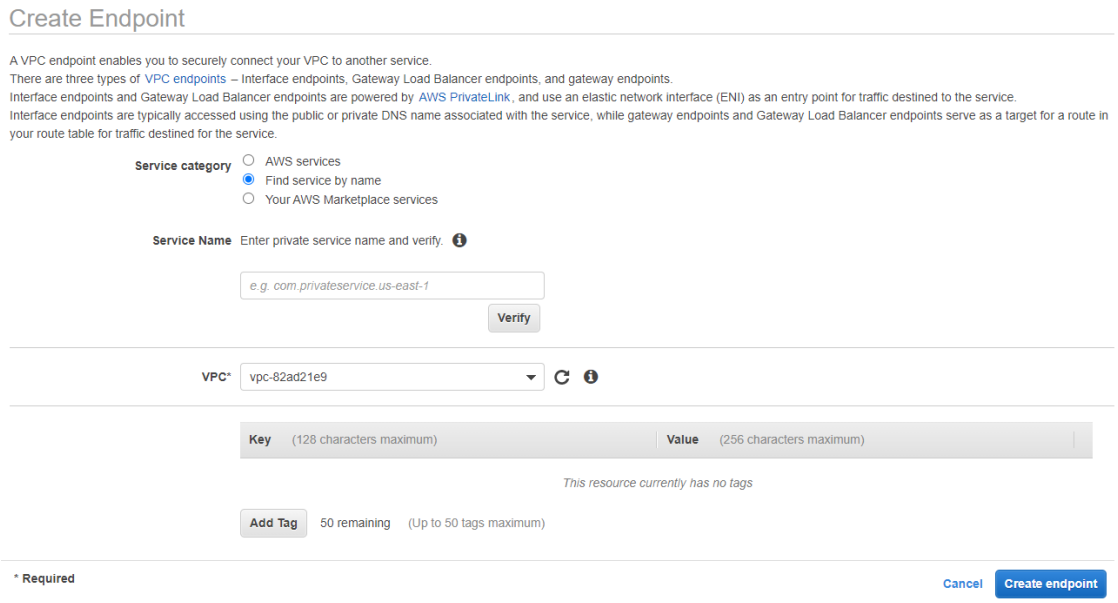

Figura 4.30 Creación del Endpoint

En la categoría de servicio, escogeremos la opción de buscar un servicio por su nombre y pegaremos el nombre copiado anteriormente. Le damos a verificar, seleccionamos nuestra *VPC* y creamos el *Endpoint*.

Antes de continuar, para poder tener estos *Endpoints* activos, nos dirigimos al servicio que hemos creado, accedemos a la pestaña de *Endpoint Connections* y aceptamos la conexión con nuestro *Endpoint*.

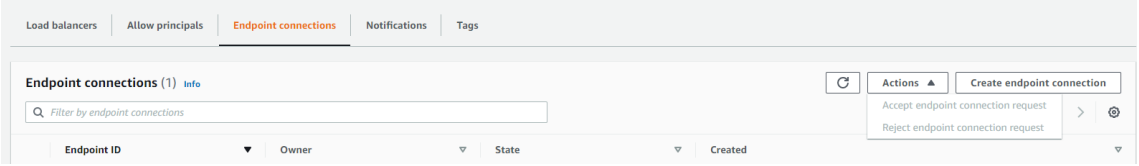

Figura 4.31 Aceptar conexión con el Endpoint

Teniendo esto hecho para las dos *VPCs*, procedemos al despliegue del *Application Load Balancer*. Empezamos creando los grupos de destino como hemo hecho anteriormente, con la diferencia de que esta vez el tipo de objetivo que tendremos serán direcciones IP en vez de instancias. Estas IPs se tendrán que especificar en el siguiente paso.

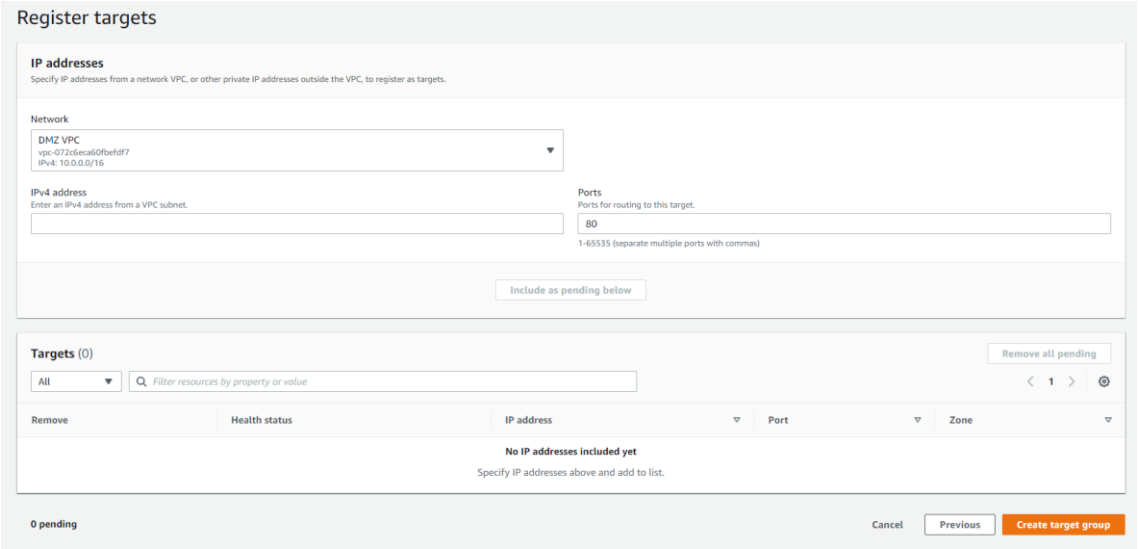

Figura 4.32 Registrar direcciones IP al ALB

La *VPC* que le corresponde es la DMZ y, las IPs que le tenemos que especificar las podremos encontrar en la pestaña de subredes que encontraremos en nuestro *Endpoint*.

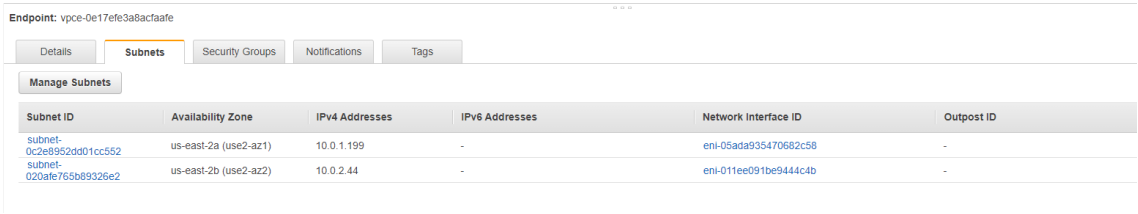

Figura 4.33 Direcciones IP que se van a registrar

Copiamos las direcciones IP que encontramos y las añadimos al balanceador de carga. Con los grupos de destino para nuestras dos redes creadas, procedemos a la creación propia del balanceador de carga, donde esta vez escogeremos el tipo balanceador de carga de aplicaciones. Le asignaremos un nombre, dejando la casilla de esquema expuesto a internet y con un agente de escucha de tipo HTTP con el puerto 80.

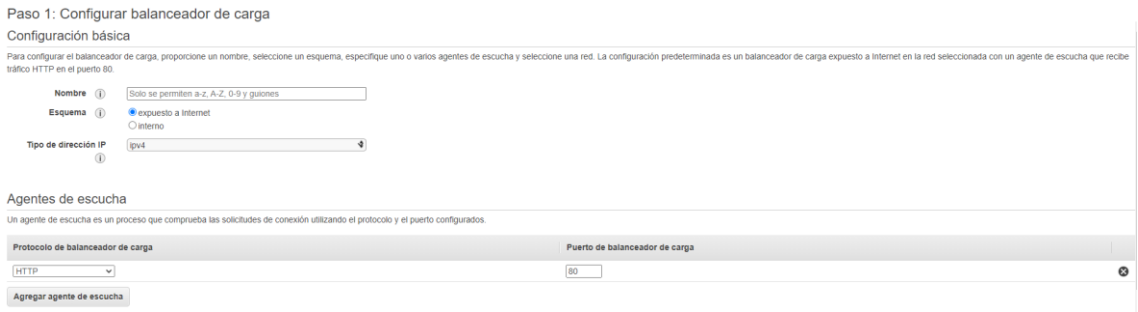

Figura 4.34 Primeras configuraciones del ALB

Para terminar esta primera configuración básica del balanceador, seleccionaremos las zonas de disponibilidad en las que vamos a trabajar.

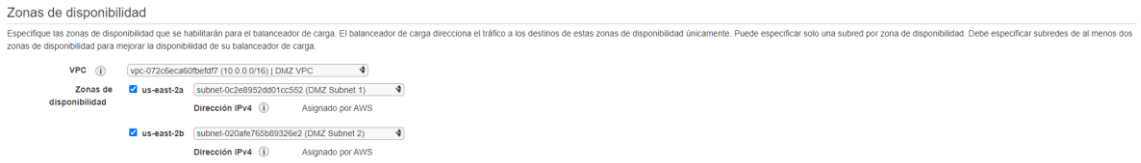

Figura 4.35 Zonas de disponibilidad para el ALB

El siguiente paso es crear un grupo de seguridad para el balanceador de carga, que tendrá permitida la entrada para el protocolo HTTP con puerto 80.

A la hora de registrar nuestros objetivos, tendremos que hacer una correcta configuración para que este funcione.

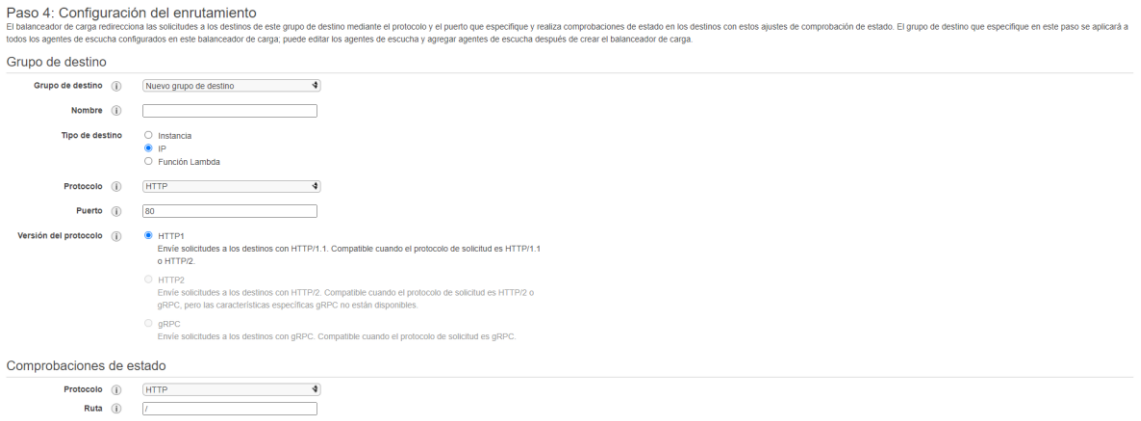

Figura 4.36 Configuración del enrutamiento del ALB

Tenemos que seleccionar el grupo destino, en nuestro caso, escogemos el grupo del servidor azul, ya que queremos que sea el por defecto de nuestro balanceador. Tendremos el tipo de destino configurado a IP y con un protocolo HTTP con el puerto 80. Para las comprobaciones, también dejaremos el protocolo HTTP.

Finalmente, registramos las IPs que tengamos disponibles.

Figura 4.37 Regitro de los servidores al ALB

Finalmente, para terminar con nuestro montaje, configuraremos nuestras rutas del balanceador de carga desde el mismo balanceador, en la pestaña de agentes de escucha. Editamos las reglas para el puerto 80, donde añadiremos nuestras rutas para redirigir al servidor azul y verde manualmente. Esto lo haremos cogiendo el DNS de nuestro *Application Load Balancer* y le añadiremos el */blue* o el */green*.

Figura 4.38 Configuración de las rutas del ALB

Antes de realizar nuestras pruebas para ver si todo funciona correctamente, tenemos que comprobar que los *Security Group* estén correctamente configurados. Para ello, han de tener las siguientes reglas:

- **SG del ALB:** acceso HTTP desde internet.
- **SG de los *Endpoints*:** acceso HTTP desde el *Security Group* de nuestro ALB.
- **SG de las instancias EC2:** acceso HTTP desde el rango de IPs del NLB propio.

4.4. Juego de pruebas

Teniendo esto, ya podemos probar cómo teniendo todas las instancias en funcionamiento, el balanceador de carga va alternando entre un servidor y otro.

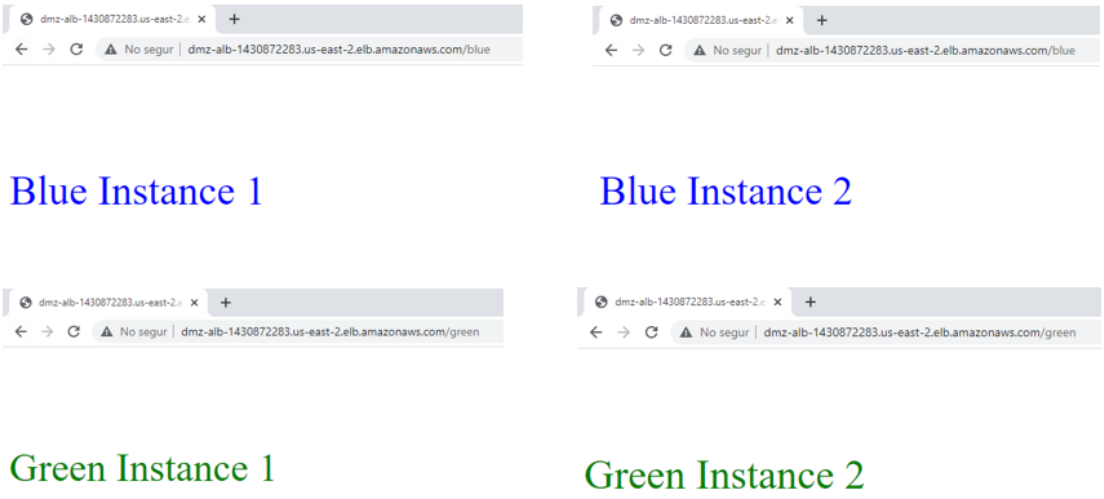

Figura 4.39 Balanceo del Application Load Balancer

Otra prueba que se puede realizar es el simular la caída de algún servidor, en este caso, vemos que únicamente nos dirige a los dos servidores que hemos dejado activos.

Figura 4.40 Prueba con dos servidores caídos

5. Conclusiones

Se ha estudiado el funcionamiento del AWS, aprendiendo cuáles son sus servicios, componentes, y cómo se deben usar.

A continuación, se ha propuesto una arquitectura segura, escalable y con elementos redundantes. Esta configuración típica no es la única posible, pero se puede considerar lo suficientemente básica para cubrir un gran número de casos o ser la base a partir de la que se obtengan sistemas más complejos.

Finalmente se ha implementado el diseño en AWS y se han realizado unas pruebas básicas de su correcto funcionamiento.

El coste aproximado de este desarrollo ha sido de unos 70 euros a lo largo de los meses de trabajo.

5.1. Trabajo futuro

El sistema implementado es perfectamente funcional, pero aun así se le podrían incorporar algunas mejoras:

- Una instalación de un WAF en los servidores EC2 le añadiría más seguridad a toda nuestra arquitectura.
- Para mejorar la escalabilidad se podría investigar alguna manera para automatizarla, ya que en este sistema se van añadiendo servidores manualmente.
- Se podría hacer configuraciones alternativas con más *VPCs* para añadirle más variedad y, a rasgos generales, mejorar más todos los puntos que buscábamos en los objetivos. Esto no se ha podido hacer ya que el número de *VPCs* gratis está limitado.

Referencias

[*1*]. (s.f.). Obtenido de https://forbes.es/empresas/53352/internet-como-medio-de-comunicacion/

[*2*]. (s.f.). Obtenido de https://ipmark.com/la-covid-19-impulsa-mayor-uso-de-internet/

[*3*]. (s.f.). Obtenido de https://www.threepoints.com/es/cloud-computing-principales-proveedores-y-casos-de-exito

[*4*]. (s.f.). Obtenido de https://docs.aws.amazon.com/