Trabajo de Fin de Grado
Localizador Gráfico de Direcciones IP

Grado de Ingeniera Informática

Autor:
Marcos Miranda Ruiz

Consultora:
María Isabel March

Fecha:
Junio de 2017

Universitat Oberta de Catalunya

© Marcos Miranda Ruiz
Reservados todos los derechos. Está prohibido la reproducción total o parcial de esta obra por cualquier medio o procedimiento, comprendidos la impresión, la reprografía, el microfilme, el tratamiento informático o cualquier otro sistema, así como la distribución de ejemplares mediante alquiler y préstamo, sin la autorización escrita del autor o de los límites que autorice la Ley de Propiedad Intelectual.

Resumen

El proyecto que se ha llevado a cabo es el de la creación de un localizador gráfico de direcciones IP. Este proyecto pertenece al ámbito de las redes de computadores y a la carrera de grado en Ingeniería informática impartida en la UOC.

Con esta herramienta lo que se pretende es que un usuario pueda localizar un dominio o IP en el mapa mundial y que de esta manera obtener información acerca de la ubicación física de aquello que se quiere buscar. La herramienta también permite la opción de realizar la búsqueda mediante traceroute, la cual es una de las bases de de este proyecto y que nos permite no solamente visualizar en el mapa la dirección final sino que también los saltos que ha dado en cada dispositivo y su ubicación correspondiente en el mapa, esto nos ayuda a visualizar la rapidez de respuesta de los contenidos de Internet y el viaje que se tiene que hacer hasta llegar a ellos.

La herramienta tiene dos partes generales que podemos distinguir claramente, una interfaz gráfica clara y fácil de manejar para el usuario que nos permitirá visualizar las consultas que realicemos y una parte de servidor que se encargara de procesar todas las peticiones y hacernos llegar la información que pidamos. Una vez nos adentremos en el proyecto explicaremos ambas partes en detalle.

Sin más preámbulos comencemos con la memoria, que espero que resulte entretenida y también cargada de conocimientos y datos interesantes.

Índice

Ilustraciones

1. Introducción

Para iniciarnos en el proyecto es bueno que tengamos una idea general de cómo funciona la geolocalización, que es y para que se utiliza, ya que en mi opinión es una herramienta muy utilizada, pero que casi pasa desapercibida a pesar de ser de gran importancia y ofrecer un alto valor añadido a distintos proyectos en numerosos campos.

La geolocalización explicada de manera sencilla consiste en detectar desde donde o hacia donde va destinada una petición, y dependiendo de la ubicación que consigamos extraer de esta dirección de origen o destino seremos capaces de tomar decisiones especificas, personalizando la respuesta para distintas regiones.

La mayoría de webs de hoy en día dispone de una herramienta de este tipo, siempre trabajando y siempre comprobando desde donde se conectan los usuarios, hasta en algunos casos almacenando esta información. La pregunta que nos puede surgir ahora es ¿Por qué? y ¿Para que nos sirve esta información?. Pues es relativamente sencillo, la respuesta pasa desde ofrecer a este usuario la pagina en un idioma u otro dependiendo desde donde sé este conectando, hasta a llegar a casos más complejos como pueden ser las limitaciones de contenido por región o estadísticas de ventas para saber en que país o región invertir más o desplegar una campaña de marketing más potente.

Con esta introducción podemos pasar a describir los pasos iniciales que se han dado para llevar a cabo este proyecto, exponiendo también las ideas y razonamientos para decantarse por una tecnología u otra.

1.1 Punto de Partida

Como ya hemos anticipado en la introducción, existen miles de casos de uso distintos para la localización mediante IP, y en realidad es una herramienta necesaria y muy actual ya que especialmente con el crecimiento de los servicios de streaming y la nube, las empresas como Netflix están invirtiendo grandes cantidades de dinero en poder detectar de donde proviene la conexión de un usuario, incluso a través de proxies, para ofrecerle el catalogo que le corresponde y ahorrarse posibles problemas de copyright.

Nosotros en este proyecto no llegaremos tan a fondo como los casos descritos anteriormente, sino que utilizaremos esta tecnología para mostrar de una manera visual y más sencilla los pasos y saltos de servidor que se producen cuando nos conectamos con algún sitio web.

La decisión por decantarme a realizar este proyecto fue tomada a raíz de que considero que es un tema muy interesante, y que sin duda es una opción para entender mejor el funcionamiento interno de cómo funcionan los componentes que utilizamos y vemos a diario.

Otro de los puntos que me llamó la atención es que yo no conocía ningún tipo de herramienta de este tipo que tuviese un aspecto visual, siempre lo había conocido como una serie de comandos a través de una terminal, una representación más que nada orientada para informáticos o personal técnico que la pudiese utilizar y entender, en este caso me gusta el reto de poder acercar esta tecnología al alcance de cualquier persona, para que la puedan probar y ver con sus propios ojos.

Para poder hacer accesible esta tecnología a todo el mundo, he pensado que la mejor forma de llevarlo a cabo seria hacerlo a través de una pagina web, ya que así todo el mundo podrá acceder a ella sin importar si quiera el sistema operativo, bastara con tener un navegador instalado. Creo que es la opción acertada ya que en los tiempos que vivimos todo se empieza a implementar en la nube y los servicios de SaaS están a la orden del día.

1.2 Objetivos

Como objetivo principal de este proyecto se quiere lograr que la herramienta diseñada pueda ser funcional, atractiva y sencilla de utilizar y que de esta manera logre localizar gráficamente en el mapa la ubicación más aproximada posible de una dirección IP o dominio. Se dará la opción al usuario de seleccionar y utilizar una funcionalidad más avanzada como lo es la opción de traceroute, la que mostrara los saltos que se han dado de servidor en servidor y al igual que la opción básica, también los representara en el mapa.

El contexto global de este proyecto se basara en las redes de computadores, siendo esta la base de el funcionamiento, pero que a su vez implicará el aprendizaje y desarrollo de tareas relacionados con la programación y diseño de paginas web, a la vez que conocimientos de arquitectura y servidores.

Algunos de los objetivos marcados que se han de cumplir son:

- Aplicación sencilla e intuitiva.
- Funcionamiento garantizado en navegadores Chrome y Firefox.
- Menor mantenimiento posible.
- Menor coste posible de desarrollo.
- Aplicación ágil y de respuesta rápida.
- Aspecto visual atractivo.

1.3 Enfoque

Antes de poder planificar el proyecto se han necesitado realizar unos pasos previos para obtener una idea general de cómo enfocar la herramienta.

He comenzado por investigar que herramientas podría utilizar y cuales serán las mejores, comparando varios aspectos como rendimiento, coste y facilidad de utilización, todo esto con la suite de Amazon Web Services (AWS) en mente, ya que quería que el proyecto fuese novedoso en cuanto a tecnologías utilizadas. Dentro de

este conjunto de herramientas se han investigado varias opciones ya que no existía solo un camino para poder realizarlo.

Desde el comienzo de el proyecto se tenia claro que tenia que existir una representación visual de la información por lo que investigue en varias soluciones de mapas y se escogió una ruta y un proveedor de mapas concreto con el que se podrían realizar las funciones necesarias para esta tarea.

Así mismo se tomo la decisión de que otros elementos o librerías utilizar que fuesen las más apropiadas para el proyecto siempre fijándonos en utilizar componentes con licencias de uso completamente gratuitas y libres de uso.

Con estas tecnologías escogidas y herramientas auxiliares estaba seguro de que podía satisfacer todas las necesidades de la herramienta y que podría construir todos los componentes que se habían previsto y especificado. Sabiendo esto pude seguir con la fase de diseño del proyecto, en la que decidí como estarían entrelazados los elementos que compondrían la herramienta en el ámbito de la arquitectura, así como los primeros diseños básicos y esbozos del aspecto global que tendría la herramienta.

Durante la fase de implementación se pusieron estas tecnologías escogidas en practica para desarrollar todas las funcionalidades que eran necesarias para llevar a cabo este proyecto. Aquí ya definiríamos dos partes fundamentales para el proyecto, una destinada para la interfaz con la que el usuario final interactuará y otra encargada de realizar toda la gestión de las peticiones.

Una vez tuviese la implementación lista, podría realizar varias pruebas para asegurarme de que el resultado que obtenemos es el que se desea.

1.4 Planificación

En este paso se describirá la planificación inicial que se realizo antes de comenzar a desarrollar la aplicación. De esta planificación inicial a la que luego se ha llevado a cabo existen pequeñas diferencias ya que se han ido encontrado problemas inesperados y que han supuesto una ligera desviación de los planes originales. A pesar de esto ha sido una buena guía para mantenerme en los plazos propuestos y establecer una carga de trabajo equilibrada a lo largo del cuatrimestre.

Tarea 1:
*Duración :*2 semanas *(5 Marzo - 19 Marzo)*

Descripción: Proceso de búsqueda de información, identificación de posibles obstáculos, escoger las mejores tecnologías y generar la idea global del proyecto:

1.1 – Tema relacionado con servidor donde crear la pagina web que acople correctamente el localizador gráfico.
1.2 – Búsqueda de automatización de desarrollo para una mayor facilidad a la hora de hacer el deploy en el servidor.

1.3 – Posibilidad de crear la pagina web basada en eventos con una arquitectura serverless.(Pensando en la suite de herramientas de AWS)
1.4 – Escoger lenguaje de programación de la pagina, donde acoplar el localizador gráfico.
1.5 – Búsqueda de opciones para realizar la representación del mapa global (Google Maps, Opciones OpenSource…)
1.6 – Obtención de la información a representar, ya sea a través de Tracert, MaxMind, whoami, DnsLookup u otras posibles soluciones.

Objetivos: Asentar las bases y escoger las tecnologías a utilizar que nos puedan dar el mejor resultado para este proyecto.

Tarea 2:
Duración : 3 semanas(19 Marzo – 9 Abril)

Descripción: Creación del servidor y proceso de automatización con repositorios:
2.1 – Creación del servidor y funcionamiento de acceso y visita desde una IP Publica.
2.2 – Automatización de servidor para que coja el ultimo código desde el repositorio.
2.3 – Asegurar el funcionamiento correcto del servidor
2.4 – Posible sincronización del Backend con alguna base de datos temporal para tener por ejemplo los resultados de las 10 ultimas IP's.

Objetivos: Tener el Backend, la parte de servidores y todas las configuraciones realizadas para ser capaz de hacer el hosting a la pagina web.

Tarea 3:
Duración : 5 Semanas (*Abril 9 – 14 Mayo)*

Descripción: Comienzo programación de la pagina web, una primera versión , con la posibilidad de utilizar un bootstrap gratuito:
3.1 – Comienzo de la programación web.
3.2 – Conseguir mostrar un mapa.
3.3 – Integrar un botón de llamada para el rastreo de IP que muestre los resultados

Objetivos: En este periodo de tiempo espero obtener una primera implementación de la pagina, muy básica, sin adornos, meramente funcional, con posibilidad de que el rastreo no se muestre todavía en el mapa

Tarea 4:
Duración : 2 Semanas (14 Mayo – 28 Mayo)

Descripción: Finalización de la pagina web, añadir detalles y asegurar el funcionamiento y el plot de las IP's en el mapa:
4.1 – Mostrar los resultados obtenidos por el botón en el mapa.
4.2 – Mejorar presentación y visualización.

Objetivos: Obtener una pagina web presentable y funcional, casi lista para la entrega.

Tarea 5:
Duración : 2 Semana *(28 Mayo – 11 Junio)*

Descripción: Repaso general de la pagina y posible solución a errores que se puedan encontrar por el camino. Este tiempo también se podrá utilizar como comodín en caso de que alguna de las tareas anteriores necesite algo más de tiempo.
5.1 – Repaso completo de la pagina
5.2 – Comprobación de funcionamiento y correcto acceso desde otros equipos
5.3 – Preparación de la entrega del producto.
5.4 – La idea es que la memoria del el proyecto se vaya completando poco a poco según se va avanzando con el trabajo para que no quede para el ultimo minuto, en estas dos semanas también se dará un repaso a la memoria, corrección de errores y completar algún detalle que pueda faltar.

Objetivos: Entrega de producto y memoria para el 11 de Junio.

Tarea 6:
Duración : 1 Semana *(11 Junio – 18 Junio)*

Descripción: Entrega de la Presentación.
6.1 – Dedicación exclusiva a la creación del video de presentación del el proyecto para su posterior entrega.
6.2 – Se utilizara la captura de pantalla QuickTime para grabar el video a la vez que se va comentando y explicando.

Objetivos: Realizar la entrega de la Presentación cumpliendo todos los requisitos exigidos.

1.5 Productos Obtenidos

Después de el desarrollo de este proyecto se espera obtener un producto en formato de pagina web, en ella encontraremos la herramienta desarrollada desde la cual seremos capaces de realizar búsquedas tanto de dominios como de direcciones IP y en cuestión de segundos obtener un resultado visual de esta búsqueda, que nos muestre información sobre la búsqueda introducida, incluida la representación gráfica en el mapa que nos ofrecerá una mejor experiencia de usuario.

En la herramienta podremos escoger varias opciones, según queramos realizar búsquedas más complejas incluyendo la opción de traceroute o sin ella.

Aquí tenemos una imagen de el producto final obtenido. Su funcionamiento y comportamiento al detalle se describirán en otros capítulos más avanzados de la memoria.

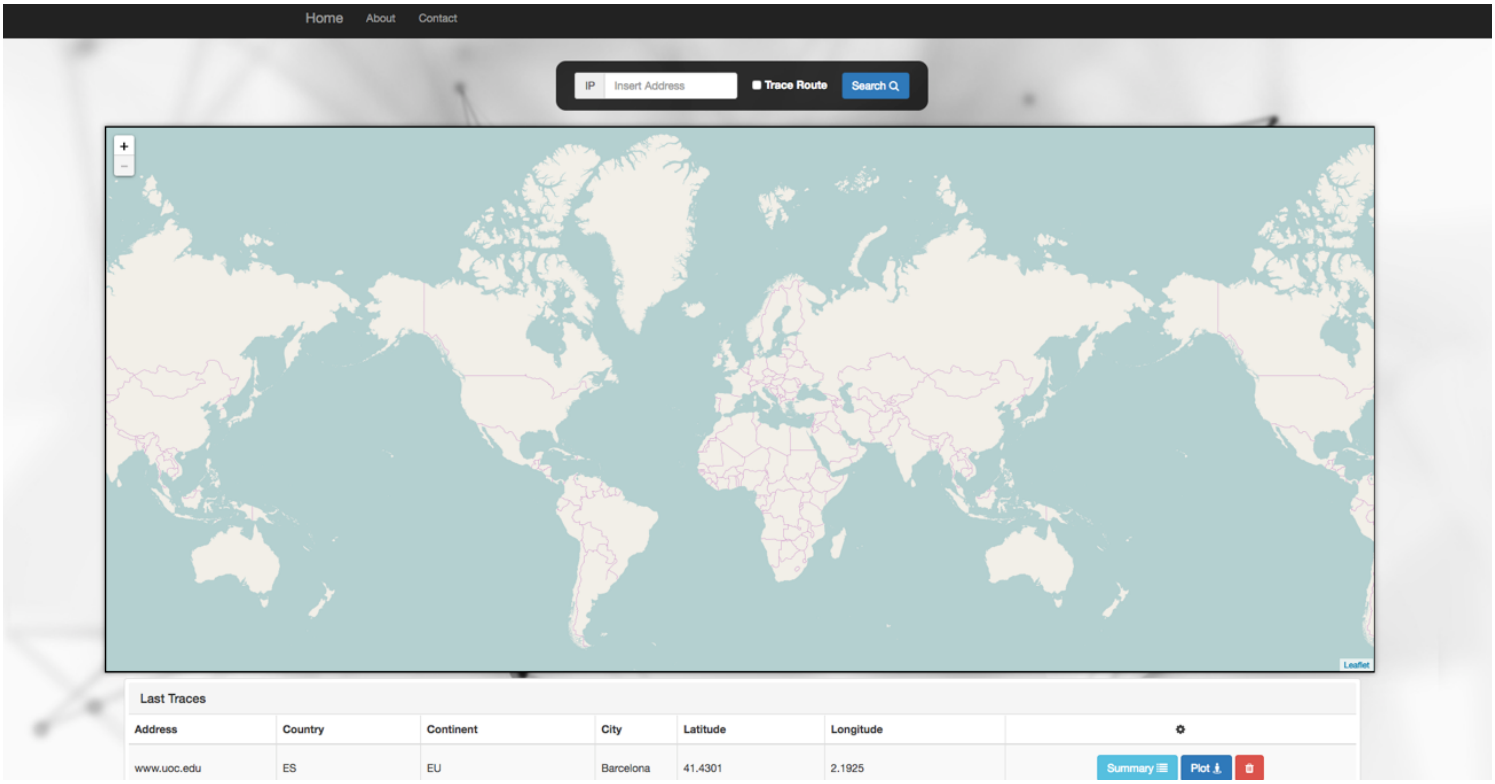


Ilustración 1- Diseño Producto Obtenido

Como podemos ver el diseño es bastante directo y sencillo, tenemos una sección para introducir la búsqueda, el mapa que representara el resultado de esta misma y una tabla que nos mostrara nuestras ultimas búsquedas así como detalles y la posibilidad de volver a ver en el mapa esta misma búsqueda sin tener que volver a realizarla.

1.6 Otros Capítulos

En el capitulo dos de esta memoria "Tecnologías y Conceptos Teóricos" describiremos un poco las tecnologías utilizadas, así como el porque de escoger precisamente cada una de estas tecnologías. Muchos de estas serán las que se utilicen durante la elaboración del proyecto.

En el capitulo tres, llamado "Implementación" encontraremos la explicación de cómo se han realizado las distintas partes del proyecto, abordando las partes más técnicas y dividiendo la herramienta en varias partes para una explicación más sencilla y modular.

En el capitulo cuatro, llamado “Pruebas” encontraremos una serie de pruebas para la demostración del funcionamiento de la plataforma es el correcto, así como una serie de comprobaciones de errores, que deberán aparecer cuando el usuario realice alguna acción incorrecta

El ultimo capitulo corresponde a las “Conclusiones“ donde se expondrás las ideas generadas a partir de la creación de esta herramienta.

2. Tecnologías y Conceptos Teóricos.

En este apartado trataremos de explicar en mayor profundidad las tecnologías y conceptos teóricos que forman la parte fundamental del proyecto.

2.1 Traceroute

Traceroute al igual que la suite de herramienta de Amazon Web Services ha sido uno de los pilares entorno al cual se ha construido el proyecto. Traceroute es una herramienta que nos permite desgranar los saltos de servidor que da una búsqueda de dirección al ser rastreada, en este apartado explicaremos en profundidad esta herramienta y su funcionamiento a nivel de protocolos de red.

En los tramos finales de elaboración de este proyecto se me comunicó que se esperaba la implementación del traceroute a nivel de sockets. Inicialmente los objetivos que definí estaban centrados en otras tareas por lo que no he alcanzado a integrar esta parte dentro del proyecto. Tras su análisis se decidió dejar este aspecto como una mejora futura, aunque esta explicada en la página 38 como se podría haber realizado además de extractos de código que simularan el funcionamiento así como las librerías necesarias de sockets y conexiones para llevar esta funcionalidad a cabo.

Comenzaremos por explicar el funcionamiento interno del traceroute y posteriormente continuaremos explicando el resto de tecnologías.

2.1.1 Funcionamiento Interno

La herramienta traceroute al comunicar datagramas y pertenecer a un sistema de comunicación abstraído en capas obedece el modelo TCP/IP. Lo que significa que como cualquier otro paquete que circula por internet posee una estructura de envío de datagramas predefinida. Deberemos entender primero esta estructura para poder continuar con la explicación interna de el traceroute. A continuación tenemos un diagrama de la estructura interna de un datagrama de los enviados por traceroute :

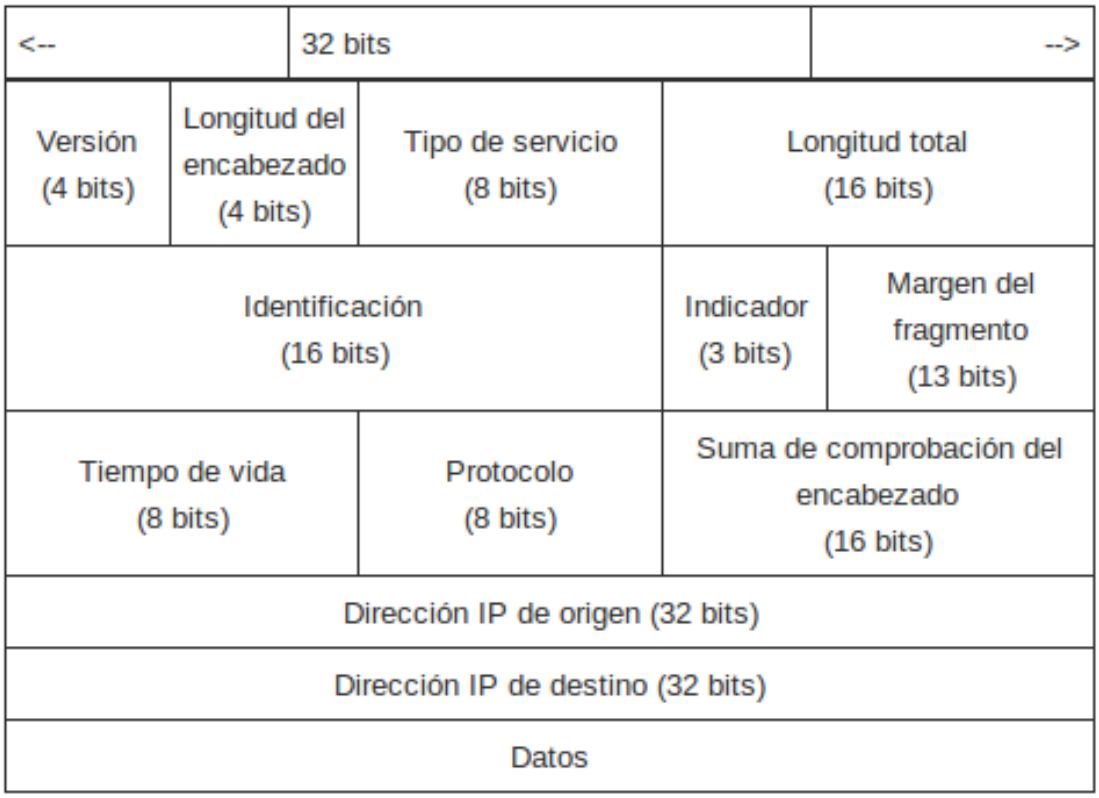


Ilustración 2- Estructura Datagrama

De este diagrama y para esta explicación, el campo que más nos interesa y del que más hablaremos es el campo de 8 bits llamado Tiempo de vida o Time to live en ingles, que a partir de ahora llamaremos TTL.

El campo TTL a pesar de llamarse Tiempo de vida no nos indica el tiempo de vida en segundos de este datagrama, sino que nos indica en máximo numero de saltos que este paquete puede dar hasta ser descartado.

Por lo que si un paquete sale de nuestro ordenador con un valor de TTL = 15 dará 15 saltos antes de ser descartado, en cada salto intermedio que se realice el valor de TTL será disminuido por uno, así hasta o bien llegar al destino final o bien que el paquete sea descartado. Cuando el paquete es descartado, el ultimo dispositivo donde se encontraba será el encargado en enviar una respuesta al la dirección de origen que leerá “ICMP TTL excedido en transito” de esta manera sabremos la dirección de el ultimo salto donde se ha quedado el paquete.

La pregunta que nos puede surgir ahora es, como sabe traceroute las direcciones de todos estos saltos intermedios cuando solo el ultimo de ellos manda una respuesta que incluye su dirección. La respuesta es que utilizamos esta funcionalidad que tiene TTL de mandar una respuesta con un mensaje para pasar individualmente por todos los dispositivos intermedios de forma que creamos un bucle que vaya aumentando gradualmente el valor de TTL de salida desde nuestro ordenador, de esta manera

sabremos cada uno de los saltos ya que todos ellos fallaran y devolverán su correspondiente respuesta, hasta llegar al final que nos devolverá un "ICMP Echo Reply", una imagen explicativa podría ser la siguiente:

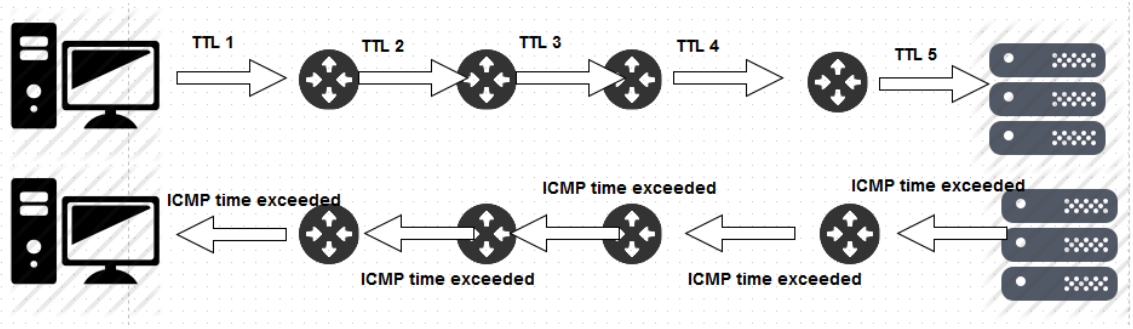


Ilustración 3- Respuesta TTL

Al ir incrementando el TTL vemos que el paquete cada vez llega más lejos y vamos formado un listado de dispositivos por los que ya ha pasado, y así seguirá hasta el numero máximo de TTL que hayamos puesto o hasta obtener una respuesta positiva.

En la explicación hasta ahora nos hemos basado en describir el concepto TTL, pero debemos saber que existen distintos tipos de traceroute que utilizan distintos protocolos para llevar a cabo la función. Durante el comienzo de la explicación y hasta ahora hemos asumido utilizar el protocolo ICMP, pero existen otras opciones que varían ligeramente de funcionamiento, aunque todas se basan en TTL.

Diferentes protocolos para traceroute:

- UDP :
  El protocolo UDP funciona de una manera muy similar al protocolo ICMP, y es que cuando agota los saltos TTL devuelve el mensaje de "TTL Time Exceeded" y cuando llega a destino final devuelve "ICMP Destination/PORT unreachable". UDP es el protocolo por defecto utilizado por el traceroute de Linux y por lo tanto el que se pondrá en uso en este proyecto.

- ICMP :
  El protocolo ICMP es muy similar al UDP descrito anteriormente, lo único que cambia es el mensaje devuelto una vez se ha llegado a destino, en este caso se devuelve un "ICMP Echo Reply". ICMP es el protocolo por defecto utilizado por el traroute (tracert) de Windows.

- TCP
  El protocolo TCP se utiliza ya que la mayoría de dispositivos entre medias nos permiten utilizar el puerto 80 utilizado para trafico TCP y esto nos asegura que los paquetes enviados no sean bloqueados por el firewall. Esta opción de traceroute se baja en mandar peticiones TCP SYN por el puerto 80.

2.2 Geolocalización

La geolocalización es una de las bases también de este proyecto y específicamente trataremos con geolocalización basadas en direcciones IP.

Esto consiste en asignar a una dirección IP una ubicación geográfica, por lo que cuando destinamos cierta IP a un dispositivo, este tiene asociado a el un país de procedencia, una región o estado, una ciudad e incluso en ocasiones información más detallada como puede ser el barrio o la calle, incluso puede contener coordenadas pertenecientes a localización GPS. Existen otras opciones de geolocalización como puede ser DNS LOC, la cual funciona de manera que cada dominio tiene asignado un parámetro LOC que indica su ubicación en coordenadas, aunque se ha optado por no utilizar esta tecnología ya que no todos los dominios tiene asignados un registro LOC.

Existen varios centros o empresas en el mundo encargados de asignar esta información que son a las que luego podemos consultar a quien pertenece ciertos dominios o podemos buscar información acerca de ellos.

Toda esta información acerca de la geolocalización de ciertas IP's está almacenada en bases de datos, existen numerosas opciones y bases de datos de las que podemos escoger para intentar geo localizar una IP. Existen algunas opciones de pago, y otras gratuitas todo dependiendo del grado de exactitud que queramos obtener, para este proyecto he optado por la opción gratuita de una de las empresas más conocidas que nos proporciona soluciones de este tipo, llamada MaxMind.

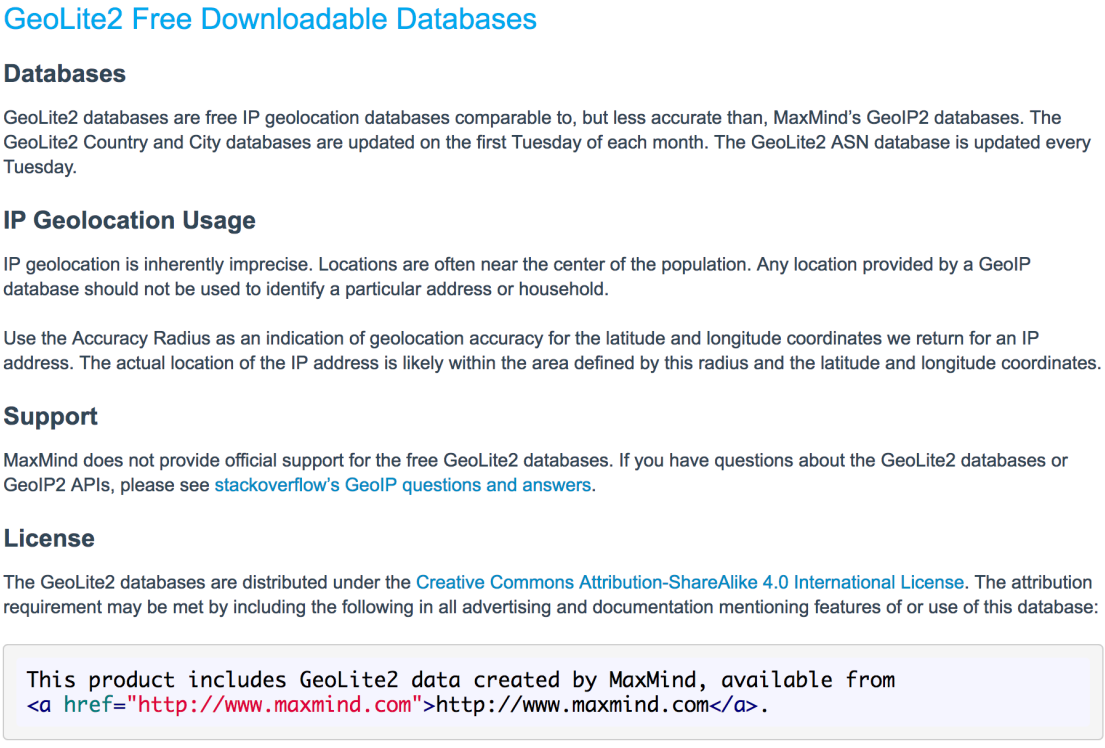
2.2.1 MaxMind

MaxMind como comentaba antes ofrece distintos grados de precisión, aquí dejo una imagen con los precios que nos ofrecen los servicios de pago y el nivel de exactitud que se puede conseguir con las búsquedas, para nuestro proyecto cualquiera de estos niveles era demasiado costoso como para poder utilizarlo, está más orientado a empresas.

Base de datos	Country	City
Aplicación	Segmente sus visitantes por país	Cuando se necesitan datos más granulares
Continente	✓	✓
País	✓	✓
Subdivisión regional		✓
Ciudad		✓
Código postal		✓
Latitud y longitud aproximadas		✓
Frecuencia de actualización	Semanal	Semanal
Cargo de la licencia del sitio (incluye un mes de actualizaciones)	$50	$370
Facturación mensual	$24 por mes	$100 por mes
Facturación anual	$314 por el primer año y $288 por cada año subsiguiente.	$1470 por el primer año y $1200 por cada año subsiguiente.
	Más información	Más información

Ilustración 4- Precios MaxMind

Para nuestro caso hemos utilizado la versión completamente gratuita de la base da datos que nos ofrece MaxMind, y que hemos acabado incorporando como librería al proyecto después de compilarla debidamente, en su pagina web existen otros ejemplos de cómo utilizar la versión gratuita de su base de datos con varios ejemplos.



GeoLite2 Free Downloadable Databases

Databases

GeoLite2 databases are free IP geolocation databases comparable to, but less accurate than, MaxMind's GeoIP2 databases. The GeoLite2 Country and City databases are updated on the first Tuesday of each month. The GeoLite2 ASN database is updated every Tuesday.

IP Geolocation Usage

IP geolocation is inherently imprecise. Locations are often near the center of the population. Any location provided by a GeoIP database should not be used to identify a particular address or household.

Use the Accuracy Radius as an indication of geolocation accuracy for the latitude and longitude coordinates we return for an IP address. The actual location of the IP address is likely within the area defined by this radius and the latitude and longitude coordinates.

Support

MaxMind does not provide official support for the free GeoLite2 databases. If you have questions about the GeoLite2 databases or GeoIP2 APIs, please see stackoverflow's GeoIP questions and answers.

License

The GeoLite2 databases are distributed under the Creative Commons Attribution-ShareAlike 4.0 International License. The attribution requirement may be met by including the following in all advertising and documentation mentioning features of or use of this database:

```
This product includes GeoLite2 data created by MaxMind, available from
<a href="http://www.maxmind.com">http://www.maxmind.com</a>.
```

Ilustración 5- Uso Gratiuto MaxMind

Una vez tengamos la librería cargada, lo único que deberíamos es llamar a los métodos correspondientes para que de una IP podamos obtener una ubicación.

2.3 DNS

Durante toda esta memoria y a lo largo del proyecto vamos a hablar muchas veces de IP's o dominios y parece que sean lo mismo y se traten de la misma manera, pero en realidad no es completamente verdad.

Es precisamente gracias a el DNS (Domain Name System) que podemos permitirnos el lujo de comparar IP's con dominios. DNS es una base de datos que almacena nombres de dominio, cada uno asignado a una IP, por eso cuando preguntamos por una dirección especifica, lo que sucede es que se accede a la base de datos que contiene los DNS y en ella se busca a que IP corresponde este nombre de dominio para poder navegar hasta él.

Cuando navegamos por internet e introducimos una dirección a la que queremos navegar no pensamos que esta dirección, una vez introducida, se traduce a una dirección IP que es la que posteriormente contacta con el servidor adecuado y nos devuelve la información asociada.

Esta tecnología también la hemos implementado en el proyecto y cuando introducimos un dominio este es traducido para posteriormente poder hacer el traceroute sobre él.

3. Implementación

El planteamiento inicial del proyecto se quiso orientar de tal manera que el producto final obtenido fuese y obedeciese una arquitectura serverless, la cual describiremos y explicaremos un poco más adelante en esta sección de implementación. Esto implicaba crear una pagina web estática, que activase eventos cada vez que un usuario interactuaba con ella.

Este primera visión duro poco ya que se encontraron algunos problemas e incompatibilidades entre la arquitectura serveless que se pretendía utilizar y las herramientas de traceroute que iban a ser la base de el proyecto, por lo que se opto por la creación de un servidor tradicional para la ejecución de las tareas de la pagina web.

Todo esto se enfoco desde un principio utilizando la suite de herramientas de Amazon (AWS), la cual ha sido también una de las piezas claves para juntar todas las partes del proyecto.

Para el desarrollo de la pagina web estática se utilizaran los lenguajes de HTML y CSS, junto con partes de JavaScript, pertenecientes a NodeJS y jQuery.

Para la parte de servidor se utilizara también NodeJS, única y exclusivamente.

Se han escogido estos lenguajes porque creo que son los que mejor se adaptan en cuanto a pesadez, ligereza y velocidad de ejecución, aprovechando que JavaScript es un lenguaje que no necesita compilado y que funciona realmente bien para tratar las respuestas de objetos de una manera más rápida y cómoda.

A continuación explicaremos las implementaciones que se han debido realizar para el conjunto global de este proyecto, analizando detalladamente el trabajo realizado.

1.1 Componentes de Terceros

Para llevar a cabo la totalidad de el proyecto se han utilizado varias fuentes, librerías y herramientas de terceros. Para todas ellas se ha asegurado que sé esta trabajando bajo la licencia MIT, que nos permite utilizar de una manera totalmente gratuita estas librerías o software. Todas ellas serán también citadas como corresponde al final de la memoria en la bibliografía, con un enlace a la fuente en cuestión. A continuación realizare un listado de los distintos componentes de terceros que he utilizado así como con una pequeña descripción de lo que han aportado al proyecto:

- AWS (Amazon Web Services)

Se han utilizado varias herramientas de la suite de Amazon para este proyecto:
  - S3: Utilizado para almacenaje y alojamiento de la pagina web estática.
  - DynamoDB: Base de datos no relacional utilizada para almacenar todas las búsquedas que se hacen a través de la aplicación y poder ofrecer el historial de ellas.
  - EC2: Instancias de servidores de diferentes tamaños y sistemas operativos, utilizada para el procesado de todo el back-end de la aplicación.
  - CloudFront: Servicio de red de entrega de contenidos globales, para acelerar la entrega de mi sitio web a todo el mundo.
- Freenom: Proveedor de dominios web donde he escogido www.marcosmirandatfg.tk para redirigir el Cloudfront y que la pagina tenga una URL de la que nos podamos acordar.
- Leaflet: Librería de JavaSript totalmente open-source y gratuita que nos servirá de mapa interactivo para mostrar las ubicaciones de las búsquedas.
- Bootstap: Framework en JavaScript que nos ayuda a desarrollar paginas web con un poco más de estilo y facilidad al utilizar filas y columnas predefinidas.
- MaxMind: Es el servicio en este caso en formato de librería que nos proporciona con una base de datos que es de donde sacaremos la ubicación y la respuesta a las búsquedas introducidas.
- Gitlab: Servicio de almacenaje de código, como cualquier otro repositorio git, que nos sirve para mantener nuestro código almacenado en la nube, de esta forma me ha sido más fácil sincronizar los cambios en el código con el servidor.
- DataTables: Plugin de jQuery que nos ayuda a mostrar la tabla de búsquedas después de recogerla del servidor o de realizar algún cambio sobre ella.
- SweetAlert2: Librería JavaScript para personalizar las alertas de mensajes de errores y de operaciones.
- Traceroute: Librería para ayuda de utilización de traceroute, que invoca al comando traceroute del sistema. Se ha utilizado para conseguir y limitar los tiempos del traceroute para que funcione de la manera más rápida posible, ya que el proceso por defecto es un poco lento.
- Particles.js: JavaScript en formato de canvas que ayuda a aplicar el estilo del puntero en la pagina de bienvenida.

1.2 Amazon Web Services

La utilización de AWS en el proyecto ha sido clave ya que al querer orientar el diseño a una pagina web, necesitaba varias herramientas, las cuales no disponía.

Mi idea inicial era utilizar una combinación de herramientas de Amazon para generar un proyecto que obedeciese la arquitectura serverless, es decir, que tuviésemos una pagina web sin ningún tipo de servidor corriendo detrás de ella. El planteamiento de esta idea generaba una gran cantidad de beneficios para el proyecto, como por ejemplo la alta velocidad que nos proporcionaría utilizar las herramientas de Amazon detrás de una VPC (Virtual Private Cloud), ya que al ser una red interna

privada, las peticiones entre la pagina web y el generador de eventos Lambda, que es la parte que nos facilitaría que la arquitectura fuese serverless, serian casi instantáneas. Otra ventaja sería el ahorro en costes y mantenimiento, ya que no existen servidores que mantener.

Explicando un poco más en profundidad esta tecnología serverless y la herramienta que nos lo proporciona, podemos decir que el servicio Lambda de la suite de herramientas de Amazon nos proporciona una opción muy novedosa de construir aplicaciones, elementos y paginas web si tener la necesidad de usar servidores. Simplemente el usuario interactuaría con la pagina web como siempre, sin embargo cuando realizase una acción, esto invocaría una Lambda (un evento) que desplegaría toda la infraestructura necesaria para llevar a cabo la operación y una vez haya concluido y retornado la respuesta, este evento desaparecería.

Como comentaba, en un inicio esta idea me parecía la mejor para llevar a cabo el proyecto, y comencé incluso a elaborarla, pero pronto me di cuenta de que existía un fallo con el que no contaba, este servicio de eventos o Lambda no me permitía ejecutar las funciones necesarias para realizar el traceroute, la cual era la base del proyecto, así que tuve que rectificar y replantearme un poco la estructura y arquitectura que debía utilizar.

Tras este fallo, pensé en cual seria la segunda mejor opción disponible con la que proceder. Tras valorar múltiples posibilidades esta resulto ser seguir utilizando una de las herramientas que utilizaba inicialmente, S3, que es un servicio de almacenamiento de nos ofrece almacenar ficheros dentro de él y a la vez nos da la posibilidad de crear una pagina web estática alojada en este mismo servicio formada por los archivos almacenados en ella. La herramienta de S3 estuvo presente en mi planteamiento inicial por lo realmente no sufrió ningún cambio y se pudo mantener.

La herramienta de S3 en el caso de alojar una pagina web estática también integra otro servicio que es indispensable para que el contenido de esta pagina web, que ha de ser publico llegue a todos los rincones del mundo y sea visible desde nuestros navegadores, este servicio se encarga de cachear contenidos, distribuirlos, y numerosas opciones más que podemos implementar como pueden ser control de regiones o bloqueo de accesos indeseados y prevención de ataques.

Este servicio se llama CloudFront, y como venia comentando es un servicio de red de entrega de contenido global (CDN) que como ventaja podemos observar que no dispone de ningún limite de uso, lo podemos usar tanto como queramos, y obviamente pagaremos por el uso que realicemos de ello, al igual que con todas las herramientas de AWS.

En la propuesta inicial era la pagina web alojada en S3 la que era responsable de contactar e invocar al servicio de lambdas, enviando unos datos y esperando otros de respuesta, en cambio ahora con esta segunda versión arquitectónica introduciremos un servidor más tradicional, que hará las funciones de lambda.

Este servidor pertenece a la herramienta también de Amazon llamada EC2, la que nos ofrece crear una serie de servidores con distintas capacidades y distintas características, según el trafico y las prestaciones que vayamos a necesitar. Yo para este proyecto he optado por la opción más pequeña posible llamada t2.micro que cuenta con 1 CPU de 2.5 GHz Intel Xeon Family y 1 GB de memoria. En esta instancia de servidor también se nos da la opción de escoger que tipo de sistema operativo queremos, desde una versión propia de Amazon Linux hasta Windows, pasando por las más conocidas como Ubuntu. Esta ultima es la que yo he escogido para alojar mi back-end y todo el código que responderá a las llamadas de traceroute y diferentes funcionalidades. En el apartado de implementación explicare más a fondo la instalación del servidor y su proceso.

Las diferencias entre la arquitectura mediante lambdas y la más tradicional utilizando servidores saltan a la vista y es que en la primera nuestra arquitectura se genera cuando es necesaria y permanece durmiente y sin ningún tipo de coste si no se utiliza, y en la segunda totalmente lo opuesto, siempre tendremos el servidor corriendo y alerta esperando llamadas para producir una respuesta, con el coste de mantenimiento que esto supone. Por eso desde un inicio decidí orientarme por la arquitectura serverless, hasta ver que no iba a ser posible.

La ultima herramienta de Amazon de la que hablaremos es DynamoDB que es una base de datos no relacional que hemos utilizado para guardar las búsquedas realizadas a través de la pagina web. A cada búsqueda realizada le estamos asignando un uuid auto generado para que cada búsqueda sea única independientemente de si lo que se ha buscado es lo mismo o no. Un uuid es un identificador único universal que se utiliza precisamente para poder diferenciar todo tipo de búsquedas o parámetros. En nuestra base de datos por lo tanto utilizaremos como clave este campo al que hemos denominado ID. Si no lo utilizásemos así no sabríamos diferenciar dos búsquedas iguales. También cada una de las búsquedas cuenta con su sello horario, que luego a la hora de mostrarlas en la tabla nos permitirá que la ultima búsqueda siempre este arriba, ordenándolas de manera descendente por hora.

Habiendo hablado de las tecnologías utilizadas dentro de la suite de herramientas de Amazon Web Services me parece interesante saber que precio y que costes tiene utilizar esta infraestructura.

Antes de comenzar a desgranar cifra por cifra los costes que tienen las herramientas utilizadas debemos saber algo muy importante y que ha sido cien por cien la razón de orientar este proyecto hacia la suite de AWS. Esta razón es que Amazon cuenta con un servicio de capa gratuita por el cual muchas de sus herramientas no tienen un coste pasado cierto gasto o consumo en datos, u otras que también se le añade el factor tiempo de poder utilizar un servicio gratuitamente hasta tal fecha. Esto me ha permitido crear el proyecto casi de forma completamente gratuita.

A continuación describiré los limites de la capa gratuita de cada uno de los servicios utilizados:

- S3 :
  - 5 GB de almacenamiento estándar.
  - 20.000 Solicitudes Get.
  - 2.000 Solicitudes Put.
  - Esta capa gratuita vence a los 12 meses a partir de la fecha de inscripción.

- EC2 :
  - 750 Horas por mes de uso de instancia t2.micro (la que se utiliza en este proyecto) para Linux, RHEL o SLES.
  - 750 Horas por mes de uso de instancia t2.micro de Windows.
  - Esta capa gratuita vence a los 12 meses a partir de la fecha de inscripción.

- DynamoDB:
  - 25 GB de almacenamiento
  - 25 Unidades de escritura y lectura (trafico de datos medido en unidades).
  - Suficiente para manejar hasta 200 millones de solicitudes al mes.
  - No vence.

- CloudFront
  - 50 GB de transferencia saliente de datos.
  - 2.000.000 Solicitudes HTTP o HTTPS.
  - Esta capa gratuita vence a los 12 meses a partir de la fecha de inscripción.

En la siguiente Ilustración podemos comprobar que en Abril, me excedí en trafico en los S3-Puts ya que fue la época en la que estaba realizando grandes cambios en la pagina web y sobrepase por muy poco él limite de la capa gratuita.

Principales servicios de capa gratuita por uso — Ver todos

Servicio	Límite de capa gratuita/uso mensual hasta la fecha	Límite de capa gratuita/uso previsto hasta el final del mes actual
EC2 - Linux	60.93% (457.00/750 Hrs)	94.45% (708.35/750 Hrs)
EBS - Volumes	32.76% (9.83/30 GB-Mo)	50.78% (15.23/30 GB-Mo)
CloudWatch - Alarms	12.34% (1.23/10 Alarms)	19.12% (1.91/10 Alarms)
S3 - Gets	5.53% (1,106.00/20,000 Requests)	8.57% (1,714.30/20,000 Requests)
S3 - Puts	2.15% (43.00/2,000 Requests)	3.33% (66.65/2,000 Requests)

Ilustración 6- Resumen Gastos AWS

Los precios a partir de la capa gratuita son muy bajos para la envergadura de este proyecto ya que miden el gasto comenzando en los primeros 50 TB/mes de consumo, lo cual para este proyecto es una exageración y nunca alcanzara esos niveles. Para que nos hagamos una idea los primeros 50 TB/mes se cobran a 0.023$ por GB de uso, lo que significa que podremos hacer uso de exhaustivo de la aplicación y quizás no llegar a pagar más de un céntimo.

Obviamente estas herramientas están pensadas para proyectos y empresas con un consumo de datos astronómico como puede ser el ejemplo de Netflix, para proyectos de final de carrera estas herramientas nos quedan un poco grandes, pero esto no quitan que sean un buena opción, además de económica para llevarlo acabo.

1.3 Pagina Web

En este apartado nos dedicaremos a la explicación de el proceso de creación de la página web, desde el diseño sencillo y apto para todos los usuarios hasta las funcionalidades más complejas que han sido desarrolladas.

1.3.1 Configuración Pagina Web

La pagina web creada es una pagina web estática almacenada en el servicio de almacenamiento de ficheros de S3, allí es donde se han subido todos los archivos pertenecientes y donde se ha configurado para que junto con el servicio de CloudFront sea una pagina visible a todo el mundo.

El la siguiente imagen podemos ver que aspecto tiene el sistema de ficheros de almacenamiento:

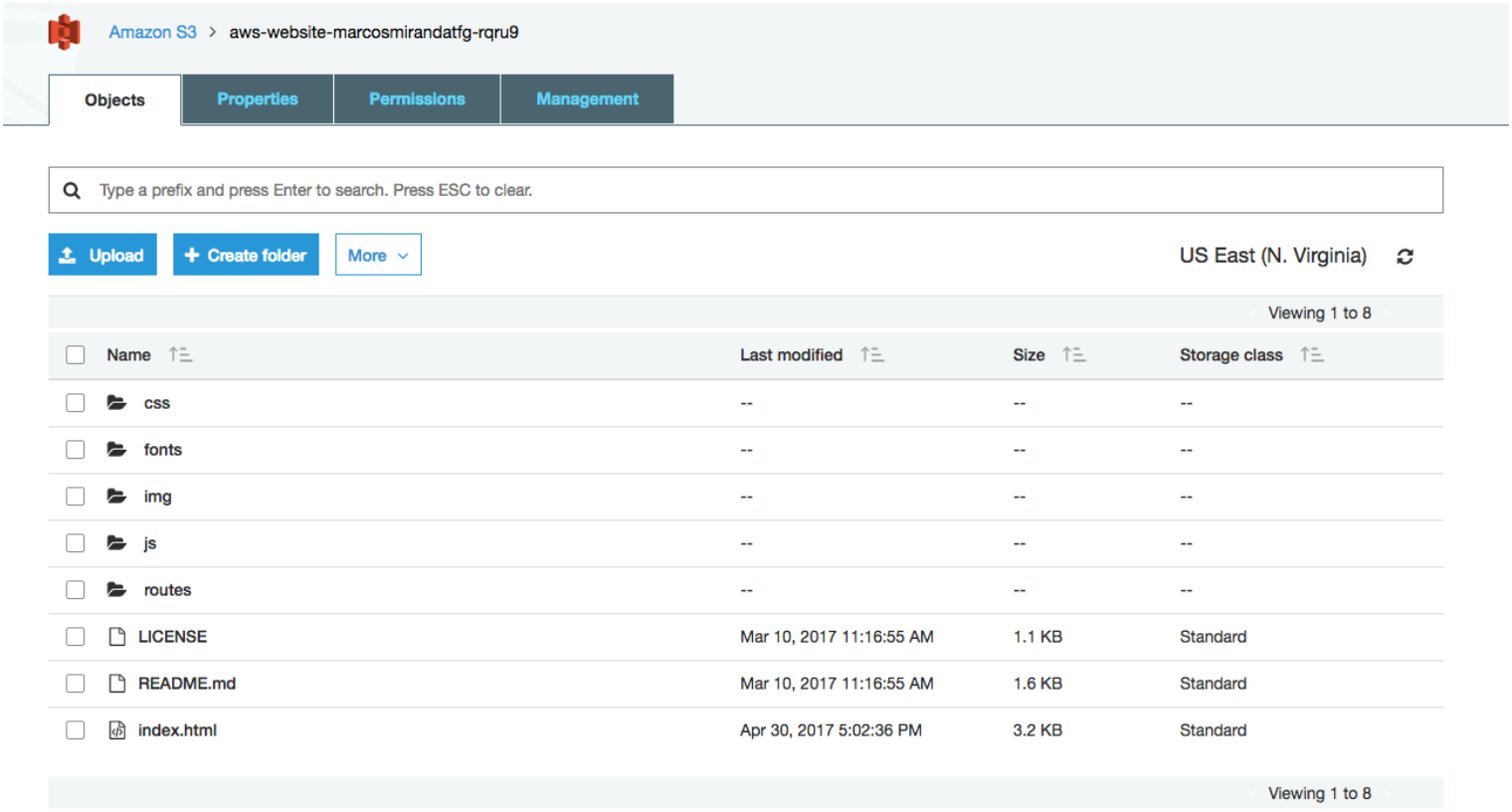


Ilustración 7- Almacenamiento S3

En la pestaña "Properties" deberemos seleccionar el index.html como archivo para que el servicio sepa en que fichero comienza la pagina web. A partir de aquí crearemos una distribución de CloudFront que será la encargada de distribuir el contenido de la pagina web por todo internet, para que todo el mundo tenga acceso. CloudFront nos permite una gran cantidad de configuraciones de las cuales realmente no necesitamos todas, sino que para este proyecto utilizaremos unas características relativamente estándar, aquí tenemos un vistazo al número de posibilidades que nos ofrece este servicio:

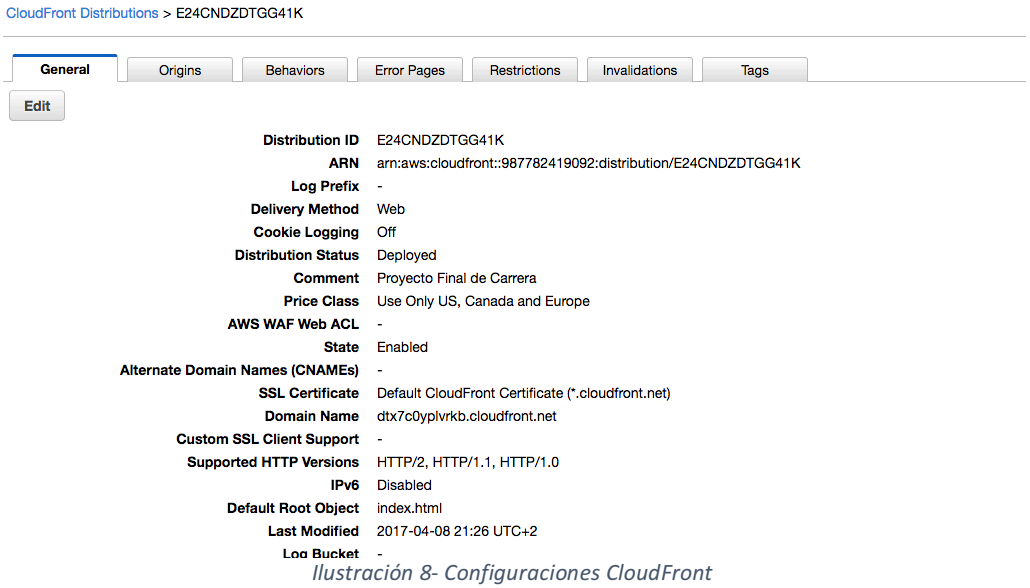


Ilustración 8- Configuraciones CloudFront

Una vez tenemos CloudFront configurado a nuestro gusto obtendremos un "domain name" que será la dirección de nuestra pagina web. Este servicio crea direcciones que son casi imposibles de acordarse para utilizar como pagina web, en nuestro caso http://dtx7c0yplvrkb.cloudfront.net, por lo que he optado por crear mi propio dominio que sea un poco más descriptivo, que redirija a este "domain name".

Para escoger el dominio, y siguiendo la tónica del proyecto de abaratar costes o incluso eliminarlos, he optado por un dominio gratuito durante un año, ofrecido por el servicio de hosting llamado Freenom, aquí he optado por escoger el dominio www.marcosmirandatfg.tk. En la siguiente ilustración muestro como redirijo el trafico de este dominio al CloudFront de Amazon:

Ilustración 9- Redirección de Dominio

Después de esto ya tendríamos una pagina web visitable y accesible desde un dominio del que fácilmente nos podemos acordar.

Al igual que para la parte de servidor, tendremos que configurar el CORS (Cross-Origin Resource Sharing) también en el servicio web para que ambas partes puedan comunicarse. Este proceso lo realizaremos tanto desde S3 como desde la distribución de CloudFront.

Prepararemos la siguiente configuración para CloudFront, como siempre añadiendo las cabeceras necesarias para la comunicación.

Ilustración 10- CORS CloudFront

Y continuaremos con el mismo proceso para S3, donde configuraremos otra vez las cabeceras necesarias.

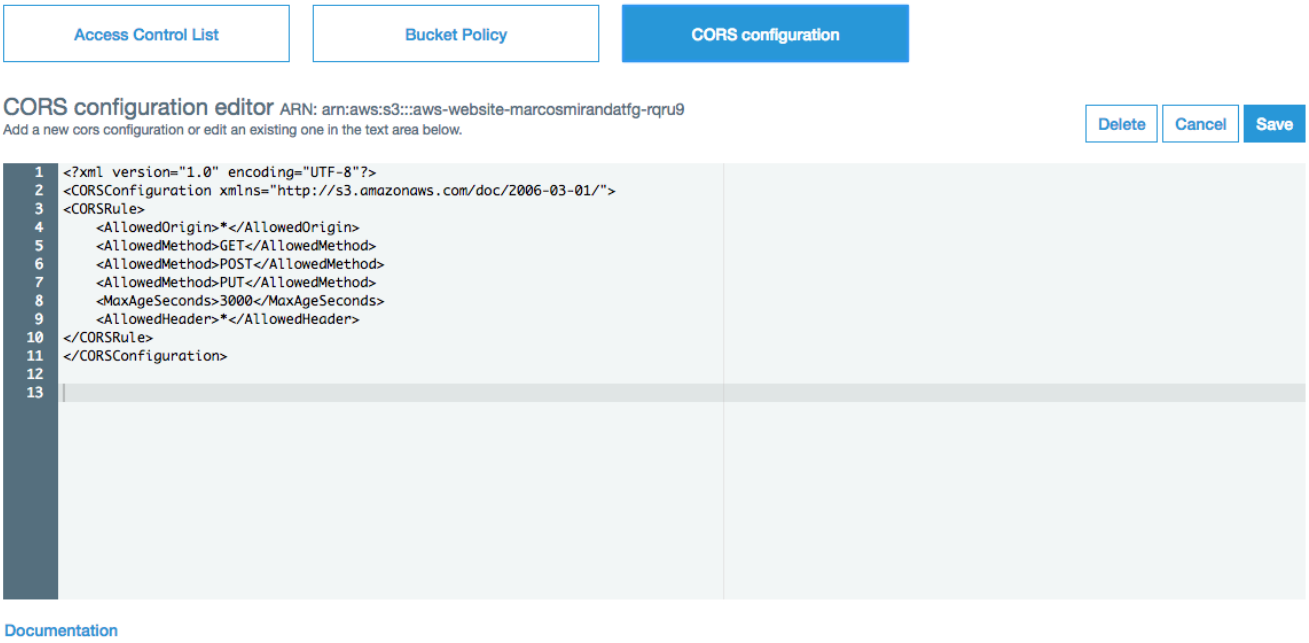


Ilustración 11- CORS S3

Con estas configuraciones ya deberíamos estar listos para poder realizar peticiones desde la pagina web, que lleguen correctamente al servidor y obtengamos una respuesta valida, aquí finalizaría el apartado de configuración de la pagina web

1.3.2 Desarrollo Pagina Web

La pagina web se comenzó a diseñar con unos primeros esbozos a papel, donde más o menos quedo claro las funcionalidades que se querían implementar y como iban a ir colocadas.

Se comenzó a desarrollar por la pagina inicial de visita, buscando la idea general y las tonalidades que se iban a utilizar para la pagina web. Poco a poco se fueron añadiendo elementos como el mapamundi del fondo, los estilos, los colores, los botones y también el canvas aplicado al cursor del ratón solo en esta página inicial que simula un poco con el movimiento los saltos de red que se producen con un traceroute.

En esta página encontraremos un botón que nos permitirá dirigirnos a la pagina principal de la aplicación.

Ilustración 12- Página Bienvenida

Una vez dentro podremos ver que el uso de la aplicación es muy intuitivo, ya que solo dispondremos de una cajita donde introduciremos la dirección que deseamos buscar y posteriormente tendremos la opción de buscar incluido traceroute o sin el. Esta es la visión general de el espacio donde introducir la búsqueda y donde se representara en el mapa.

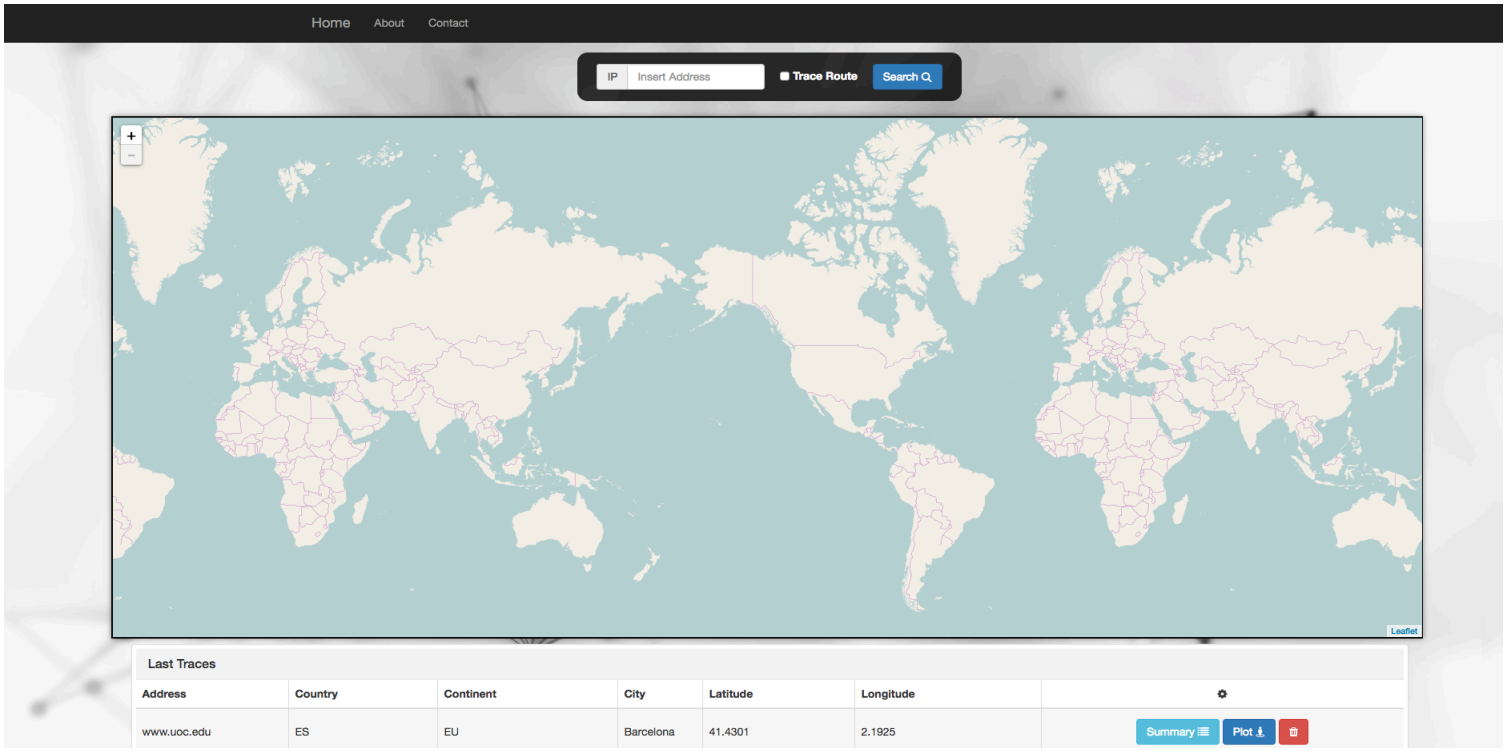


Ilustración 13- Página Principal

A parte de representar correctamente las búsquedas tenemos unas cuantas funcionalidades añadidas como que cuando se realice una búsqueda esta quedara registrada en nuestra base de datos y podremos verla almacenada en la lista “Last Traces” al igual que podemos ver la búsqueda de www.uoc.edu en la ilustración anterior.

En la tabla de “Last Traces” podremos hacer numerosas operaciones con una búsqueda anterior, desde ver la información directamente correspondiente a esa búsqueda en la misma tabla hasta distintas funcionalidades que nos permites los botones incorporados a cada una de las filas. Comenzaremos a describir sus funciones de derecha a izquierda.

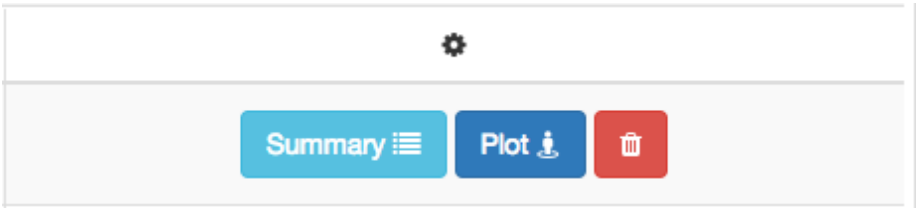


Ilustración 14- Zoom Botones

Comenzaremos con el icono que representa a una papelera, este icono claramente elimina esta búsqueda de la base de datos, eliminando completamente la fila y toda la información asociada a esta búsqueda.

El siguiente botón que lee “Plot” se encarga de volver a representar de nuevo una búsqueda realizada con anterioridad, es decir, si queremos volver a ver en el mapa la ubicación de cierta dirección haremos click en este botón, y el solo se encargara de centrarnos en el mapa y mostrarnos el resultado representado que buscábamos.

El ultimo botón que lee “Summary” ha sido sin duda el más complejo de implementar, teniendo en cuenta que si hacemos una búsqueda incluyendo la opción de traceroute, la información que obtenemos es extensiva, de cada salto tendremos coordenadas, país, continente, cuidad y otros datos y si suponemos que el traceroute realiza quince saltos, tendremos quince veces esta información de cada uno de los saltos. Representar esto de una forma que fuese visualmente comprensible ha sido extremadamente complicado, pero al final he optado por representarlo en formato JSON formando un árbol donde cada nodo se pueda expandir o contraer según la información que busquemos. Para búsquedas también he incluido una caja explícitamente para esto, donde podremos introducir lo que busquemos y el árbol se encargara donde podemos encontrar esa información subrayando en rojo el camino que hemos de seguir.

En las siguientes imágenes podemos ver como seria el resultado representado en modo árbol, el la primera imagen, con el resultado de la búsqueda contraído y en la segunda expandido, cada nodo que siga en color azul sigue conteniendo información dentro de el por lo que se podrá expandir aún más.

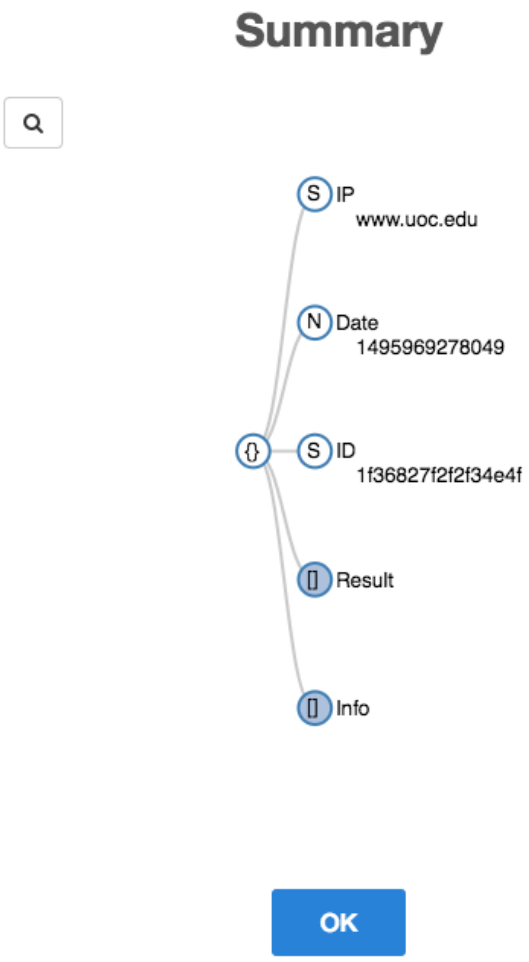


Ilustración 15- JsonTree sin expandir

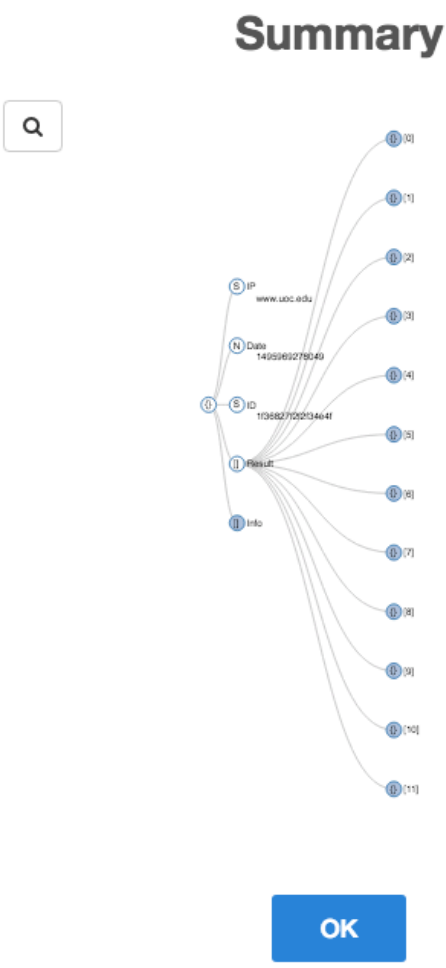


Ilustración 16- JsonTree expandido

Si queremos hacer una búsqueda dentro del árbol, simplemente deberemos dar un click al botón de la lupa y escribir lo que buscamos, si el campo está repetido dentro del árbol podremos avanzar o retroceder con el uso de las flechas y el árbol se ira abriendo por los nodos correctos que contengan eso que buscamos.

Una representación de búsqueda podría ser la siguiente :

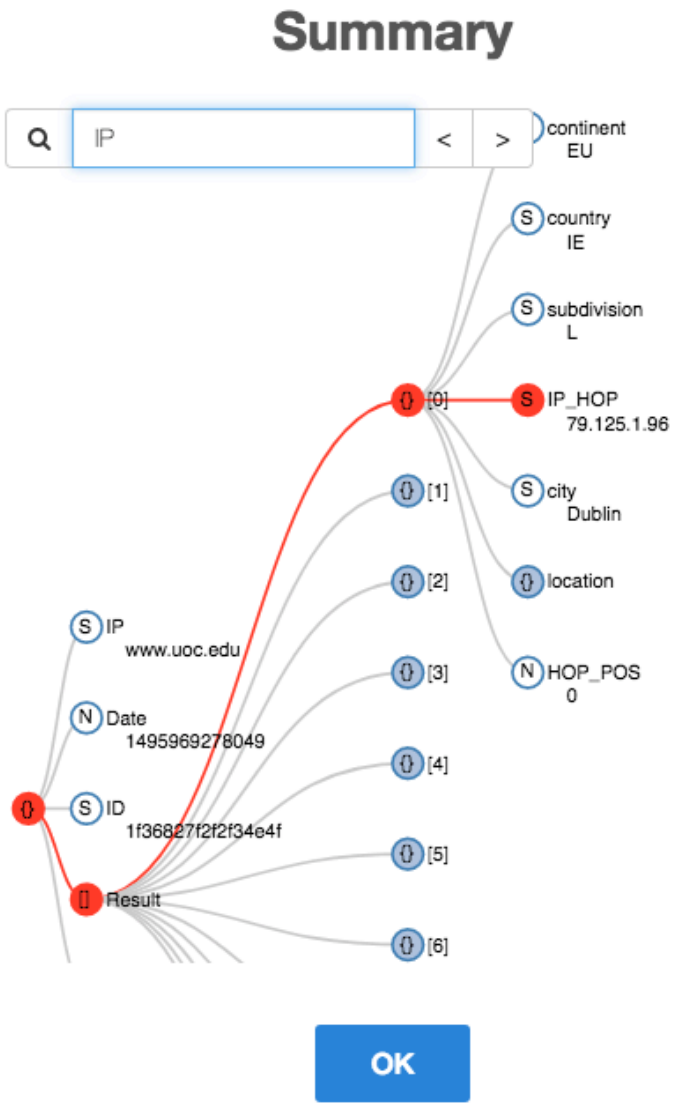


Ilustración 17- JsonTree Búsqueda

Con esta herramienta podremos leer de manera cómoda toda la información que se ha obtenido, sin tener que realizar una tabla muy compleja. Es posible que de ciertas búsquedas queden campos en blanco según la efectividad de la búsqueda, esto es normal ya que no podemos controlar la exactitud de la geolocalización, dependiendo de en que país se busque, existe más información almacenada en la base de datos que en otros.

A parte de estas funcionalidades, en el mapa podremos ver información sobre los saltos que se realizan para llegar hasta el destino si hemos seleccionado la opción de traceroute, en cada uno de los marcadores que veremos en el mapa nos indicara que numero de salto es y que dirección de IP tiene ese salto. En el salto final podremos ver toda esta información incluyendo también el texto End, indicando que la búsqueda traceroute ha terminado aquí, aquí tenemos como se representaría una búsqueda con traceroute en el mapa.

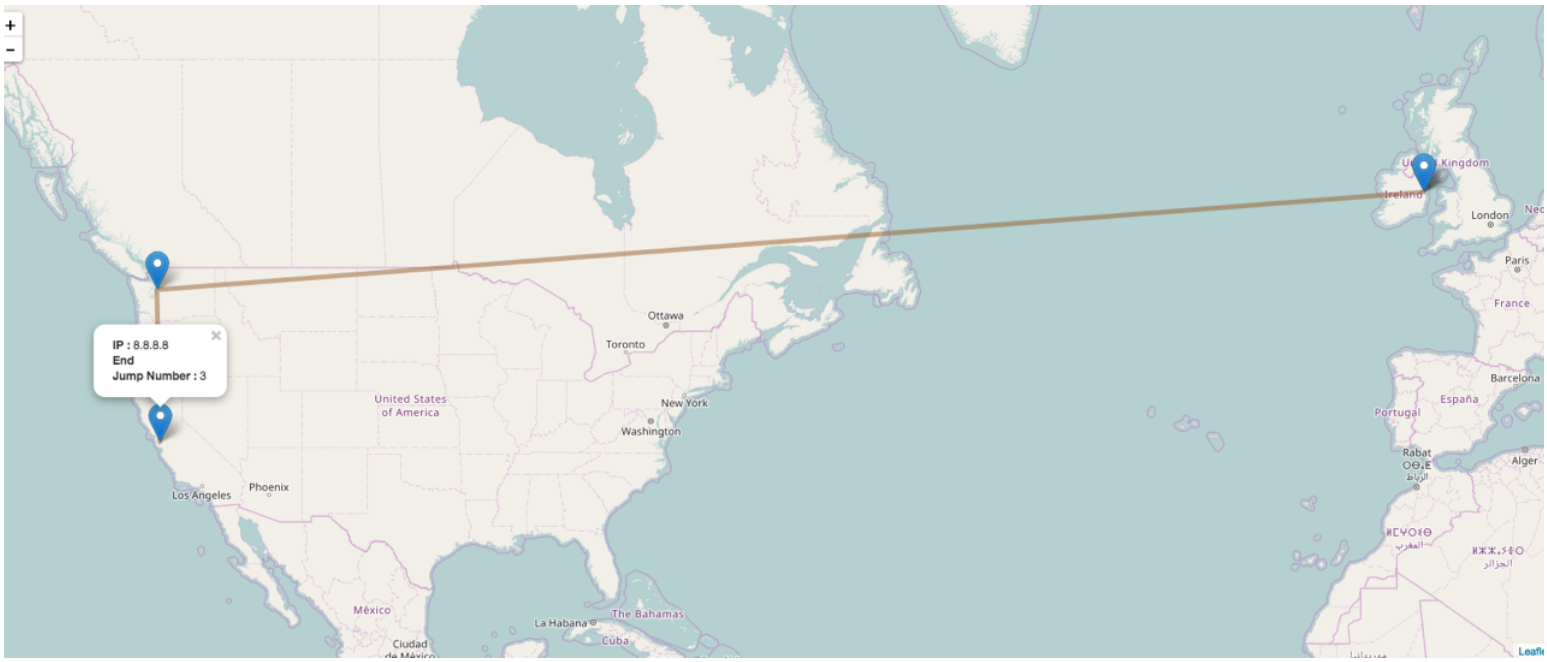


Ilustración 18- Representación Busqueda Traceroute

Como comentábamos antes, si nos interesa saber la ubicación de un salto intermedio, no tendremos más que hacer click sobre el pequeño icono, copiar la IP correspondiente a este salto, y en el resumen de la búsqueda (“Summary”) buscar esa IP, para poder ver a que ciudad, país y conteniente pertenece.

A parte de esta página donde se encuentra la funcionalidad completa de la herramienta también tendremos dos pestañas más en nuestra pagina web, una que será el “About”, que nos dará un poco de información acerca de que es esta página y porque se ha creado, donde también adjuntare la memoria y otra que se llamara “Contact” donde dejara un poco de información acerca de mi y como contactarme.

1.4 Servidor

Comenzaremos con la explicación de la creación del servidor que nos va a gestionar las peticiones realizadas por la pagina web, este servidor se encargara de realizar las funciones de traceroute, así como todas las comprobaciones de si una direcciones introducida es valida o si ha ocurrido un error en la ejecución. También se encargara de establecer la conexión con la base de datos para tanto añadir como eliminar elementos.

La creación del servidor fue a través del servicio de instancias EC2 de Amazon donde se selecciono la opción gratuita de t2.micro con el sistema operativo Ubuntu.

En la siguiente imagen podemos ver las instancias que tenemos creadas y las que están corriendo, la llamada Ubuntu es la que será nuestro servidor y la que se llama AWS-LINUX es una instancia que hemos utilizado para compilar correctamente la

base de datos de geoIP2 que nos proporciona el servicio de ubicación de las direcciones introducidas :

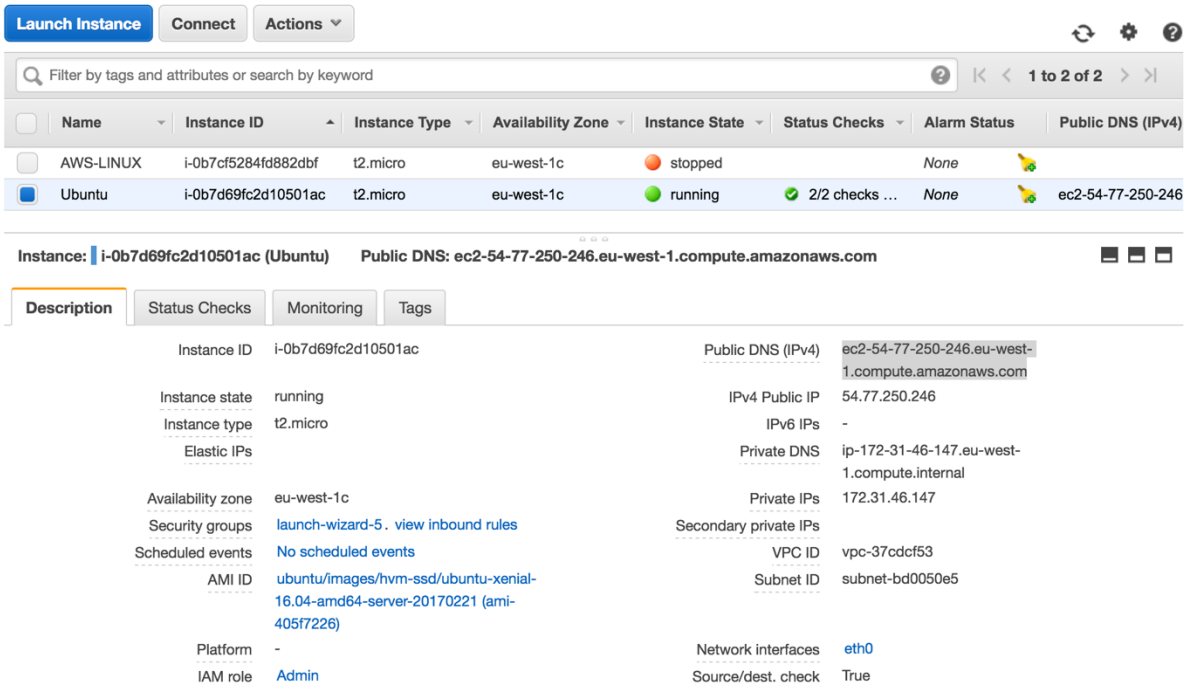


Ilustración 19- Visión Instancias EC2

Mediante una conexión ssh a la dirección subrayada en la imagen nos conectamos al servidor y comenzamos a instalar todo lo necesario para que pueda responder a nuestras peticiones.

```
MacBook-Pro-de-Marcos:Proyecto marcosmiranda$ ssh -i Marcos-Proyecto.pem ubuntu@ec2-54-77-250-246.eu-west-1.compute.amazonaws.com
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.4.0-72-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  Get cloud support with Ubuntu Advantage Cloud Guest:
    http://www.ubuntu.com/business/services/cloud

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


*** System restart required ***
Last login: Sat Apr 29 17:08:15 2017 from 83.57.19.5
```

Ilustración 20- Conexión SSH

Empezamos con las cosas fundamentales y básicas como instalar NodeJS que será nuestro lenguaje por defecto, continuamos con otras herramientas como PM2, que lo utilizaremos de servidor local corriendo en el puerto 3000 de manera interna, otras herramientas pueden ser el repositorio git, para tener nuestro código organizado y seguro, el cual podremos actualizar con simples comandos y por último y casi la herramienta más importante un Nginx, que es un servidor que utilizaremos de servidor web proxy conectado a nuestro PM2 interno.

A continuación procederé a detallar un poco más la explicación acompañándola de pantallazos para enseñar todas las configuraciones que se han realizado, comenzare en orden e iré explicando desde el exterior hacia dentro hasta llegar a la parte más interna del servidor.

La parte más exterior del servidor es el propio Nginx, que actúa de servidor y de proxy a la vez y que una vez configurado correctamente se encargara de transmitir nuestras peticiones hacia componentes más internos.

```
server {
  listen 80;
  server_name tutorial;
  location / {
    proxy_set_header   'Access-Control-Allow-Origin'    '*';
    proxy_set_header   'Access-Control-Allow-Methods'   'GET, POST, OPTIONS';
    proxy_set_header  X-Real-IP  $remote_addr;
    proxy_set_header  Host       $http_host;
    proxy_pass        http://127.0.0.1:3000;
  }
}
```

Ilustración 21- Configuración Nginx

Todas las peticiones que lleguen serán recibidas por el Nginx por el puerto 80 (determinado en la configuración por el "listen 80") que es el puerto de comunicación normal utilizado para la comunicación HTTP, una vez la petición haya sido recibida, Nginx actuara de proxy y redirigirá esta petición que le ha llegado y la reenviara por el puerto 3000, que es un puerto interno en nuestro localhost. Este puerto 3000 esta activo de manera interna ya que gracias a PM2 podemos arrancar nuestro código y generar un pequeño servidor interno, para ser más precisos la clase app.js es la encargada de arrancarse y crear este servidor interno gracias a la librería express. Podemos ver claramente que esta clase de lo único que se encarga es de crear este mini servidor y de repartir las peticiones al código correspondiente según lo que se pida.

app.js

```
const express = require('express')
const app     = express()
const ip      = require('./routes/ip');
const table   = require('./routes/table');
const cors    = require('cors')


app.use(cors())
app.use('/ip'    , ip);
app.use('/table' , table);
app.listen(3000, () => console.log('Server running on port 3000'))

module.exports = app

```

Ilustración 22- App.js

Al tratarse de un proyecto pequeño, la utilización de express para definir las pocas rutas que existen quizás sea excesivo, pero esto nos permite una fácil estructuración si el proyecto creciera más de lo esperado. Simplemente habría que añadir nuevas clases para definir rutas que proporcionasen nuevas API's a las que la pagina web pudiese llamar para obtener información.

Otro de los puntos claves del servidor y del proyecto en global ha sido la funcionalidad CORS (Cross-Origin Resource Sharing), necesaria para poder establecer la comunicación entre la pagina web y el servidor. Quizás sea este punto donde más problemas he podido encontrar, ya que para establecer este tipo de conexión he debido adaptar y agregar nuevas cabeceras a las llamadas HTTP tanto en el proxy (como podemos ver en la Ilustración 6), como al utilizar express y obviamente configurando esto mismo en la parte de la pagina web que mostraremos más adelante.

Una posible solución hubiese sido utilizar solamente el protocolo HTTPS, configurando certificados para su uso, además de adaptar el resto del proyecto ya que cambiaríamos del puerto 80 al puerto 443 comúnmente utilizado por este protocolo. Esto nos permitiría obtener un cifrado SSL de la comunicación y hubiese solucionado todos los problemas aparecidos por el CORS, ya que son motivos precisamente de seguridad los que nos proporcionaban estos problemas.

Como funcionalidad implementada dentro del servidor podemos diferenciar dos métodos principales: El primero, que se encarga de la geo localización de una dirección sin retornar información acerca del traceroute, simplemente nos llevara al destino final de donde se encuentra la dirección introducida representada en el mapa. La segunda, que es la más compleja, es la que se encarga de realizar la función completa del traceroute, llamando para cada salto de servidor a la función de geo localizar, para poder mostrar cada punto posteriormente en el mapa.

Además de las anteriores existen otros métodos auxiliares que se encargan del resto de funciones que podemos ver en la pagina web, desde llamadas de inserción,

borrado y listado a la base de datos hasta funciones de comprobación de que la dirección introducida en la búsqueda es ciertamente una dirección valida y formada de una manera correcta.

Todo el código se subía al servidor y se actualizaba mediante repositorios GIT, una vez se subían cambios de el código a GitLab, desde el servidor se realizaba una actualización y se descargaban los nuevos cambios, de esta manera siempre tenemos los cambios realizados organizados y no perdemos código ya que el propio repositorio nos sirve como copia de resguardo. La siguiente imagen muestra como está compuesto nuestro repositorio en Gitlab:

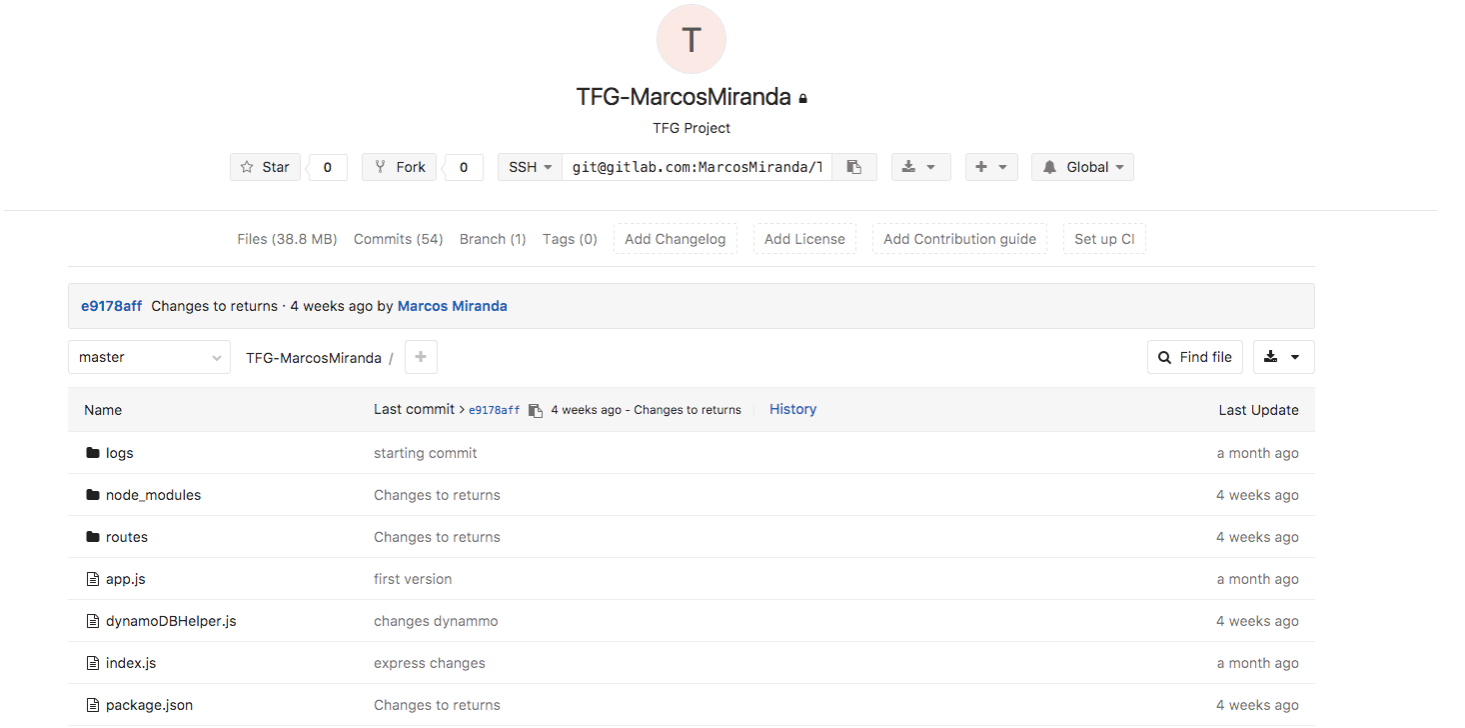


Ilustración 23- Repositorio Gitlab

1.5 Traceroute Propio

En este apartado explicaremos un poco como podría haber sido la creación de nuestro propio traceroute, aplicando los conocimientos que hemos demostrado en el apartado anterior.

Lo primero que necesitaremos hacer en utilizar la librería que nos permita la conexión mediante sockets, en NodeJS podremos usar raw-socket, y la podremos encontrar en https://www.npmjs.com/package/raw-socket, al igual que el resto de librerías de este proyecto, está publicado bajo la licencia totalmente libre de uso MIT.

A partir de aquí el usuario podrá introducir una dirección de búsqueda, si esta dirección se introduce en formato URL utilizaremos la librería dns-lookup para convertir esta URL a una dirección IP tal y como se realiza ahora mismo con el traceroute por defecto.

Para realizar la función de traceroute, necesitaremos una función ping, y posteriormente una función que realice un bucle con estos ping y que con cada iteración el valor TTL aumente para que vayamos conociendo los dispositivos intermedios que existen entre desde donde realizamos la búsqueda hasta donde se encuentra nuestro destino.

El código seria el siguiente:

```
1
2   // cargamos Librerias
3   const raw    = require('raw-socket');
4   const lookup = require('dns-lookup');
5
6
7   var MAX_HOPS = 15;
8   var TIME_LIMIT = 5000;
9
10  // esta funcion sera la encargada de hacer cada ping individual dentro del traceroute
11  function ping(destino, ttl, callback)
12  {
13      // abrimos sockets y configuramos
14      var buffer = new Buffer([]);
15      var socketLevel  = raw.SocketLevel.IPPROTO_IP;
16      var socketOption = raw.SocketOption.IP_TTL;
17      var socket = raw.createSocket();
18      // a partir de aqui aplicaremos el tipo de protocolo al socket segun nos interese
19      socket.setOption(socketLevel,socketOption, ttl);
20      socket.send(buffer,0,buffer.length,destino,function(err,resultado)
21      {
22          if(err) console.log(err);
23      });
24      // si el socket contiene una respuesta devolvemos el mensaje
25      var timeout = true;
26      socket.on("message",function(buffer,source)
27      {
28          socket.close();
29          timeout = false;
30          callback(buffer,source,time);
31      });
32
33      // aqui controlaremos si la resupesta excede el tiempo respuesta
34      // para que el ping no se quede colgado
35      setTimeout(function()
36      {
37          if(timeout) callback(null,'???',TIME_LIMIT);
38      },TIME_LIMIT);
39  }
40
41  //funcion que se encarga de realizar el traceroute recursivamente hasta finalizar
42  function traceroute(destino, ttl)
43  {
44      ping(destino,ttl,function(buffer,source,time)
45      {
46          //comprobamos si es el ultimo salto o nos hemos pasado de saltos
47          if(source == destino || ttl = MAX_HOPS)
48              process.exit(0); // si es el caso salimos
49          //si no continuamos con la recursividad de la funcion traceroute
50          traceroute(destino, ttl+1); //aumentamos el TTL
51      });
52  }
53
```

Ilustración 24- Código Traceroute propio

4. Pruebas

En esta sección comprobaremos que el funcionamiento de la herramienta creada es el que se desea y que todos los componentes funcionan como se espera.

1.1 Pruebas de Dominio

Las primeras pruebas que podemos realizar es al intentar introducir una dirección no valida en la casilla de búsqueda.

Prueba : “google.es”
Respuesta : (Solo repetiremos la imagen del error esta vez)
Resultado : Correcto

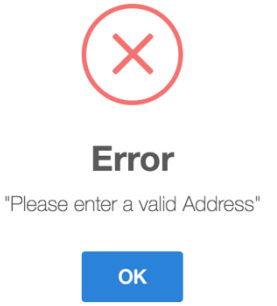


Ilustración 25- Imagen Error

Prueba : “wwww.google.es”
Respuesta: “Please enter a valid Address”
Resultado : Corrrecto
Prueba: “www.google”
Respuesta: “Please enter a valid Address”
Resultado : Corrrecto
Prueba “www.google.ok”
Respuesta : “Error looking up the domain: www.google.ok”
Resultado : Correcto

1.2 Pruebas de Traceroute

Estas pruebas consisten en asegurarse de que las búsquedas que se realizan retornan un resultado esperado. En este caso las pruebas son un poco más complejas y las he decidido hacer por terminal de comandos y con el código ejecutándose localmente para poder probar de una manera más rápida.

Después de la revisión y alguna de las pruebas de la consultora he visto que algunas direcciones fallaban al intentar hacer el traceroute y he tenido que investigar porque.

Resultaba que algunas de las direcciones que se introducían estaban siendo convertirdas a ipv6 en vez de ipv4. De esta manera el programa de traceroute era incapaz de localizar la IP de origen y la búsqueda fallaba.

La ipv6 tenia este formato:

```
traceroute -q 1 -m 20 -w 1 2a00:1450:400b:c03::5e
```

Ilustración 26- Búsqueda ipv6

Y
con ipv4 la misma búsqueda tenia este formato:

```
traceroute -q 1 -m 15 -n -w 1 130.206.13.20
```

Ilustración 27- Búsqueda ipv4

Después de arreglar el error y realizar varias pruebas de ejecución con distintos dominios se volvió a dar por bueno el funcionamiento, aquí adjunto alguna prueba mas del resultado obtenido:

```
traceroute -q 1 -m 15 -n -w 1 193.110.128.109
[ { '192.168.144.1': [ 3.027 ] },
  { '81.46.131.77': [ 5.125 ] },
  { '80.58.81.46': [ 7.041 ] },
  { '94.142.103.177': [ 4.157 ] },
  { '94.142.117.69': [ 36.61 ] },
  { '213.140.39.197': [ 36.805 ] },
  { '80.231.62.58': [ 43.532 ] },
  false,
  { '80.231.48.30': [ 46.67 ] },
  { '80.231.48.6': [ 41.242 ] },
  { '72.52.60.194': [ 40.356 ] },
  { '72.52.60.205': [ 39.803 ] },
  { '209.200.162.204': [ 37.122 ] },
  false ]
traceroute -q 1 -m 15 -n -w 1 213.73.40.242
[ { '192.168.144.1': [ 22.925 ] },
  { '80.58.64.181': [ 12.662 ] },
  { '80.58.106.161': [ 11.536 ] },
  { '193.149.1.26': [ 12.41 ] },
```

Ilustración 28- Revisión Traceroute

Aquí podemos ver como ahora si el programa esta convirtiendo todas las direcciones correctamente con lo que podremos realizar correctamente el traceroute.

5. Conclusiones

Una vez concluido este trabajo puedo decir con toda seguridad que he mejorado mi conocimiento sobre los protocolos TCP/IP a nivel de capa de red y capa de transporte y también de el conocimiento de cómo funciona la geolocalización y como implementarla.

Otro de los aspectos muy valiosos que también he desarrollado durante este proyecto es el uso de la suite de herramientas de AWS, he obtenido muchos conocimientos acerca de las herramientas utilizadas así como de las que no he podido utilizar y los problemas que me han surgido con ellas.

Creo que en general se han conseguido todos los objetivos marcados y el producto final obtenido tiene una presencia y una funcionalidad adecuadas, todos los añadidos que se quisieron incluir desde un principio han podido ser añadidos a tiempo y también funcionado como esperado, añadiendo gran valor a la aplicación.

La planificación inicial ha servido de guía para marcar los tiempos que se habían estimado para realizar cada una de las tareas, esto ha ayudado a organizar el tiempo disponible y aunque se han encontrado imprevistos, como los comentados a lo largo de este proyecto, se ha seguido más o menos la planificación establecida.

Como líneas de futuro trabajo me gustaría mejorar el proceso de traceroute y su representación en el mapa, ya que puede ocurrir que varias IP's del traceroute queden representadas en la misma zona, solo mostrando un marcador en el mapa y reemplazando los anteriores, se debería implementar cierta lógica en el código para que esto no suceda y se puedan representar todos los saltos significativos.

Con esto damos el proyecto por concluido, en general ha sido una experiencia enriquecedora de la cual se han aprendido muchos nuevos conocimientos entre ellos a gestionarse uno mismo un proyecto y cumplir con los tiempos marcados.

6. Glosario

AWS: Suite de herramientas de Amazon Web Services que se han utilizado para llevar a cabo este proyecto.

ICMP: ICMP es el sub protocolo de control y notificaciones de errores del protocolo de internet IP.

IP: Dirección IP, el numero que identifica a cada dispositivo dentro de una red

TRACEROUTE: Herramienta que nos permite seguir la pista de los paquetes que vienen desde un punto de la red.

DOMINIO: Un dominio de Internet es un nombre único que identifica a un sitio web en Internet.

GIT: Es un software de control de versiones pensado en la eficiencia y el control de versiones de código.

MIT (licencia): La licencia MIT permite reutilizar software dentro de software propio.

CORS: Cross-Origin resource sharing es el mecanismo que nos permite acceder a recursos bloqueados en una pagina web desde otro lugar web.

HTTP: Hypertext Transfer Protocol es el protocolo de comunicación que permite las transferencias de información en la web

HTTPS: Hypertext Transfer Protocol Secure es un protocolo basado en http pero que implementa una capa de seguridad para la transferencia segura de datos.

API: Las API son un conjunto de comandos, funciones y protocolos que permiten desarrollar y crear distintos programas con una facilidad añadida.

PM2: Controlador de procesos avanzados para Node.js que nos permite crear un servidor localmente

JSON: Es un tipo de formato de objeto con unas propiedades muy definidas y muy útil para el tratamiento de datos complejos.

VPC: Es un conjunto de recursos configurables bajo demando que están ubicados en el interior del cloud.

7. Bibliografía

Información sobre MaxMind:
https://www.maxmind.com/es/home

Canvas de Partículas:
https://tympanus.net/codrops/2014/09/23/animated-background-headers/

Imagen Mapa:
http://es.freeimages.com/download/file/2c1ed3629fe3ac15e81733c792fd962b/1920x960, http://es.freeimages.com/photo/global-map-1444310

Fondo geométrico
http://www.wallup.net

Imagen Datagrama:
http://www.ebah.com.br/content/ABAAAgalcAH/tp3-ruteo-dinamico-ospf-sobre-equipos-cisco

Información sobre TTL:
http://www.slashroot.in/how-does-traceroute-work-and-examples-using-traceroute-command

Información sobre Traceroute:
https://en.wikipedia.org/wiki/Traceroute

Información sobre HTTPS:
https://es.wikipedia.org/wiki/Hypertext_Transfer_Protocol_Secure

Información sobre CORS:
https://developer.mozilla.org/es/docs/Web/HTTP/Access_control_CORS

Información sobre DNS:
https://www.xatakamovil.com/conectividad/como-funciona-internet-dns