FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS

IMPLEMENTACIÓN DE UN SISTEMA PARA LA ADMINISTRACIÓN DE LAS FUNCIONES BÁSICAS DE LOS EQUIPOS DEL DATA CENTER ACADÉMICO DE LA UNIVERSIDAD DE LAS AMÉRICAS MEDIANTE CISCO DEVNET Y AMAZON ALEXA

AUTORES

Jorge Andrés Guevara Valdez

Matheo Nicolás López Zumárraga

AÑO

2020

FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS

IMPLEMENTACIÓN DE UN SISTEMA PARA LA ADMINISTRACIÓN DE LAS FUNCIONES BÁSICAS DE LOS EQUIPOS DEL DATA CENTER ACADÉMICO DE LA UNIVERSIDAD DE LAS AMÉRICAS MEDIANTE CISCO DEVNET Y AMAZON ALEXA

Trabajo de Titulación presentado en conformidad a los requisitos establecidos para optar por el título de Ingenieros en Electrónica y Redes de Información.

Profesor Guía

Mg. Ángel Jaramillo

Autores

Jorge Andrés Guevara Valdez

Matheo Nicolás López Zumárraga

Año

2020

DECLARACIÓN PROFESOR GUÍA

“Declaro haber dirigido el trabajo, Implementación de un sistema para la administración de las funciones básicas de un UCS mediante Cisco DevNet y Amazon Alexa en el data center académico de la Universidad de las Américas, a través de reuniones periódicas con los estudiantes Jorge Andrés Guevara Valdez y Matheo Nicolas López Zumárraga, en el semestre 202010, orientando sus conocimientos y competencias para un eficiente desarrollo del tema escogido y dando cumplimiento a todas las disposiciones vigentes que regulan los Trabajos de Titulación”.

_____

Ángel Gabriel Jaramillo Alcázar

Director de la Carrera de Ingeniería Electrónica y Redes de Información

CI:1715891964

DECLARACIÓN PROFESOR CORRECTOR

“Declaro haber revisado este trabajo, Implementación de un sistema para la administración de las funciones básicas de un UCS mediante Cisco DevNet y Amazon Alexa en el data center académico de la Universidad de las Américas, de los estudiantes Jorge Andrés Guevara Valdez y Matheo Nicolas López Zumárraga, en el semestre 202010, dando cumplimiento a todas las disposiciones vigentes que regulan los Trabajos de Titulación”.

William Eduardo Villegas Chiliquinga
Doctor en Informática
CI: 1715338263

DECLARACIÓN DE AUTORÍA DE LOS ESTUDIANTES

“Declaro que este trabajo es original, de nuestra autoría, que se han citado las fuentes correspondientes y que en su ejecución se respetaron las disposiciones legales que protegen los derechos de los autores vigentes”.

Jorge Andrés Guevara Valdez	Matheo Nicolás López Zumárraga
CI: 1718582610	CI: 1725371684

AGRADECIMIENTOS

Quiero agradecer en primer lugar a Dios quien a lo largo de mi vida me ha bendecido y guiado por el buen camino. También quiero agradecer a mi familia quienes me han apoyado y dado ánimos para cumplir mis objetivos. En cuanto a mi formación les agradezco a mis profesores por las enseñanzas y conocimientos que me impartieron en sus clases en especial a mi tutor Ángel Jaramillo por su gran amistad y apoyo incondicional. Finalmente, quiero agradecer a mis amigos tanto de España como de Ecuador por acompañarme en este camino y conseguir finalizar esta importante etapa.

AGRADECIMIENTOS

Agradezco a Dios primeramente por bendecirme a lo largo de este camino y permitirme culminar este objetivo. A mi madre Alexandra Zumárraga y familia por ser un apoyo incondicional en mi vida, a los docentes de la Facultad quienes con sus consejos y enseñanzas me han ayudado en esta etapa. Especialmente a mi tutor Ángel Jaramillo quien con su experiencia, motivación y conocimiento me oriento en este trabajo. Finalmente, a mis amigos que se convirtieron en hermanos que muy pronto serán mis colegas, gracias por todo su apoyo y diversión.

DEDICATORIA

Mi tesis la dedico con todo mi amor y cariño a mis padres Nancy Valdez e Iñigo Gutiérrez por su sacrificio y esfuerzo que pusieron en mí. A mis abuelos en especial a Ángel Valdez y Alberto Gutiérrez quienes siempre me apoyaron y lo siguen haciendo, aunque no estén aquí conmigo. A mi tía Ana Valdez quien es como una segunda madre para mí y a mis hermanos quienes son lo más preciado que tengo.

DEDICATORIA

Dedico este trabajo con todo amor y cariño a toda mi familia, pero especialmente a mi abuelito Jaime Zumárraga que siempre estará conmigo. Y a todas las personas que me apoyaron a lo largo de este trabajo.

RESUMEN

Hoy en día, los avances tecnológicos son cada vez de gran magnitud. Los usuarios buscan nuevos sistemas que les faciliten las cosas en sus trabajos y que sean de fácil interpretación.

La automatización de los sistemas solventa muchas de las demandas por parte de los usuarios. Por este motivo se presenta una solución basada en la implementación de un sistema para la administración de las funciones básicas de un sistema informático unificado (en inglés, *Unified Computing System* o UCS) mediante Cisco Devnet y Amazon Alexa en el centro de datos académico de la Universidad de las Américas.

Lo que se pretende conseguir con este proyecto de titulación es brindar al administrador de la red, una forma de trabajo más fácil. Podrá realizar configuraciones en el UCS mediante comando de voz que los recibe el asistente Alexa a través del Amazon Echo.

Ya que al estar configurado con una dirección IP pública la Universidad de las Américas proporcionará seguridad a nivel de red. El administrador podrá efectuar sus configuraciones desde su hogar, ofreciendo otra alternativa de modelo de trabajo.

La idea es que se desarrollen más sistemas para personas que presentan alguna discapacidad y con esta nueva temática, se pretende ayudar a dichas personas. En el caso de una persona que no tiene extremidades superiores podrá realizar el trabajo de administrador de red, ya que mediante comandos de voz podrá realizar diferentes configuraciones.

Palabras Clave: automatización, red, dirección IP, UCS.

ABSTRACT

Today, technological advances are becoming great. Users are looking for new systems that make things easier for them in their respective jobs, as well as being easy to interpret.

System automation meets many of the demands by users. Therefore, a system-based solution is presented for managing the basic functions of a unified computer system (UCS) using Cisco Devnet and Amazon Alexa in the data center University of the Americas.

What is intended to be achieved with this title project is to provide the network administrator with an easier way of working. You can make configurations in the UCS using voice command that is received by the Alexa wizard through Amazon Echo.

Since it is configured with a public IP address, the University of the Americas will provide security at the network level. The administrator will be able to make their configurations from home, offering another work model alternative.

The idea is that more systems will be developed for people who have a disability and with this new theme, it is intended to help these people. In the case of a person who does not have upper limbs, they will be able to perform the work of network administrator, because using voice commands will be able to make different settings.

Keywords: automation, network, IP address, UCS.

ÍNDICE

1. Capítulo I. Introducción

En la actualidad, los avances tecnológicos se enfocan cada vez más en el desarrollo, innovación y automatización. Dichos avances permitirán mejorar la administración de la red consiguiendo así redes más adaptables en cuanto a su infraestructura, como es el caso de Cisco DevNet que nos ofrece todas estas facilidades.

“Cisco DevNet, una plataforma creada en el año 2013 que otorga a los desarrolladores las herramientas, recursos y códigos necesarios para crear y construir soluciones innovadoras y viables para la red.” ("DevNet, red de desarrolladores de Cisco crece en Latino América", 2015). Además de ayudar a los desarrolladores y profesionales que se encuentran en el área de TI. “Esta plataforma ofrece diferentes posibilidades a los usuarios como software, seguridad, centro de datos, *IoT, cloud* y desarrollo de código abierto, que con el rápido avance tecnológico estas nuevas tecnologías serán el futuro.” ("Cisco DevNet: APIs, SDKs, Sandbox, and Community for Cisco Developers", 2019).

La forma tradicional que se usa en la configuración y administración de un data center es mediante comandos, por este motivo se decidió implementar un sistema para automatizar las funcionalidades básicas de un UCS utilizando Amazon Alexa, Amazon Web Services y Cisco DevNet. Este sistema será implementado en el data center académico de la Universidad de las Américas.

“Las redes definidas por software permiten a las organizaciones automatizar mediante la implementación y distribución de aplicaciones. Las redes definidas por software (en inglés, *Software Defined Networking* o *SDN*) incrementan los beneficios de la virtualización del centro de datos, ya que aumentan la flexibilidad, utilización de recursos, reducen los gastos generales y los costos de infraestructura.” (Ariganello, 2017).

“La biblioteca del kit de desarrollo de software (en inglés, *software development kit* o SDK) de *Python* resume algunos de los detalles de bajo nivel y proporciona interfases de programación de aplicaciones (en inglés, *application programming*

interface o API) simples para administrar UCS. Por lo que es más fácil automatizar la configuración y el monitoreo utilizando las abstracciones proporcionadas por el SDK. Proporciona *API's* que permiten configurar, eliminar, comparar y agregar objetos gestionados." ("Automatización de Cisco UCS usando *Python* SDK", 2014).

"Amazon Echo es un producto que salió al mercado en el año de 2014, es un altavoz de característica cilíndrica el cual sirve como herramienta para hacer factible el manejo de un asistente artificial en este caso Alexa, que se basa en la voz." (Gallardo Vázquez, 2019).

1.1 Objetivo General

Implementar un sistema para la administración de las funciones básicas de un UCS mediante Cisco DevNet y Amazon Alexa en el data center académico de la UDLA.

1.2 Objetivos específicos

- Determinar las funcionalidades básicas realizadas por un administrador de un UCS.
- Examinar los componentes y equipos que conforman el data center académico de la Universidad de las Américas.
- Desarrollar el sistema por medio del lenguaje de programación *Python*, el cual permitirá ejecutar los comandos dentro del UCS Manager y vincularlos al sistema Amazon Alexa.

1.3 Alcance

El alcance de este proyecto de titulación es implementar un sistema para la administración de las funciones básicas de un UCS mediante Cisco DevNet y Amazon Alexa en el data center académico de la Universidad de las Américas.

Esta solución será diseñada para ayudar y facilitar a los usuarios encargados de la administración de un UCS.

Para llegar a cumplir con lo mencionado anteriormente, el sistema será desarrollado en el UCS académico que se encuentra en la Universidad de las Américas el cual se encontrará vinculado al UCS Manager.

En cuanto a la programación, será realizada mediante el lenguaje de *Python* que permite realizar diferentes codificaciones para ejecutar consultas, configurar, crear funciones y máquinas virtuales. Estos comandos se vincularán al sistema Amazon Alexa y se usará el dispositivo de control de voz Amazon Echo, para reconocer y sincronizar los comandos, de manera que permita administrar el UCS.

Se van a realizar pruebas de funcionamiento del sistema en el data center académico de la Universidad de las Américas.

Para alcanzar el cumplimiento de lo mencionado anteriormente se va a utilizar lo aprendido en las materias de: redes y programación.

1.4 Justificación

Debido a los nuevos avances tecnológicos lo que se busca es que los sistemas sean más autónomos ofreciendo mayor facilidad para administración y configuración, es por esta razón se propone este trabajo de titulación.

Por medio de este sistema se obtendrá una mayor facilidad para gestionar el UCS académico que se encuentra en la Universidad de las Américas, permitiéndole al administrador de red reducir el tiempo que emplea al momento de ejecutar comandos por medio de la consola o interfaz gráfica, ya que lo haría utilizando el dispositivo Amazon Echo.

Este trabajo de titulación puede ser la base para gestionar nuevos proyectos que se enfoquen en temas relacionados a networking, seguridad de redes, IoT

(Internet of Things), Cloud, Centros de Datos y Meraki, utilizando asistentes virtuales.

Considerando este proyecto para una situación compleja, puede llegar a beneficiar a personas con discapacidad de las extremidades superiores o discapacidad visual, ya que podrían configurar y obtener datos por medio del sistema Amazon Alexa vinculado a la herramienta Amazon Echo.

2. Capítulo II. Marco Teórico

2.1 Asistentes Virtuales

En este capítulo se tratarán diversas plataformas y herramientas para el desarrollo de este trabajo de fin de carrera. Basándose en las consultas de trabajos relacionados con este proyecto de investigación, se observó que no existe gran variedad de casos de estudio, dado que antes no se tenía mucho conocimiento de Cisco DevNet y los recursos que esta plataforma brinda.

Se puede encontrar en esta plataforma una gran variedad de laboratorios según las necesidades del usuario, permitiendo desarrollarlos e interactuar con ellos. Gracias a esta plataforma surgió la idea de esta propuesta de tesis, ya que se encontró un laboratorio que consiste en vincular diferentes plataformas como Amazon y Cisco.

Es por ello que John McDonough con su aporte a la comunidad de *DevNet Creations Beta*, que mediante el uso de *UCS PowerTool* y la combinación de las habilidades de Amazon Alexa, lo que llegó a pensar es que con el uso del dispositivo de Amazon Echo se lograría preguntar el estado del servidor y que este mismo le responda por medio de voz el estado en el que se encuentra. Para lograrlo lo que combino fue herramientas de Amazon como: *Lambda, S3, CloudWatch* y VM de Amazon y en cuento a Cisco el UCS PowerTool. (McDonough, n.d.).

Otro proyecto detalla cómo implementar y desarrollar otro tipo de solución con el asistente virtual Alexa y por el lado de Cisco utiliza Meraki APIs que por medio de comandos de voz permite obtener su administración y monitoreo. Esto fue detallado por Juan y Uriel en Cisco Live, otro sitio que permite a los desarrolladores de Cisco DevNet dar apoyo a la gente que se proyecta en este campo y que en la actualidad está tomando más fuerza. (Castilleja & Arriaga Alvarado, 2018).

Sin duda alguna Alexa tiene una gran variedad de habilidades que permiten obtener un mayor alcance, como es el caso de Cisco WebEx, usado para videoconferencias programadas. Esto lo detalla Muthu Somasundaram en su publicación. (Somasundaram, 2018).

Aun así, se puede concluir que no existe mucha información para este tipo de proyecto en específico, pero esto no quita que sean proyectos futuristas y con un gran potencial gracias a la integración de asistentes virtuales como Alexa de Amazon. Por este motivo, en la actualidad la empresa Cisco se está enfocando y priorizando este tipo de desarrollos innovadores.

Además del dispositivo de Alexa de Amazon, existen otros tipos de asistentes de voz como *Google Assistant, Cortana, DuerOS, Siri, Bixby y Watson*. La mayor parte de estos asistentes han sido utilizados en proyectos basados en la domótica, pero ninguno de ellos se ha centrado en el campo de redes ya sea por sus restrictivas políticas en el caso de alguno de ellos o simplemente no se han realizado ningún intento. Sin embargo, hay una excepción con el asistente de voz Watson de IBM, el cual fue usado como dispositivo intermediario entre el usuario y el sistema desarrollado para la “recopilación de preguntas y respuestas médicas usando reconocimiento de lenguaje natural sobre un sistema cognitivo” (Guevara & López, 2016).

Asistente de voz	Año de lanzamiento	Desarrollador
Siri	2011	Apple
Cortana	2014	Microsoft

Alexa	2014	Amazon
Google Assistant	2016	Google
Bixby	2017	Samsung
Watson Assistant	2018	IBM

Tabla 1. *Asistentes Virtuales*

2.2 Cisco DevNet

DevNet es una plataforma que tuvo origen en el año 2014 gracias a Susie Wee, una persona experta en tecnología, ofreciendo gran variedad de herramientas, recursos, códigos y requerimientos que permiten innovar y desarrollar nuevas soluciones dentro del área de redes automatizadas.

En esta plataforma se encuentran una gran cantidad de laboratorios y repositorios de códigos, los cuales permiten practicar, aprender e investigar en varios temas como:

- *Cloud Computing.*
- *Internet of Things (IoT).*
- *Networking.*
- *Data Center.*
- *Security.*
- *Meraki*
- *Webex.*

En la tecnología que se va a centrar este proyecto de implementación será en Data Center específicamente en un Cisco UCS.

2.3 Cisco Unified Computer System (Cisco UCS)

2.3.1 Cisco UCS Manager

Cisco UCS Manager permite administrar de mejor manera gracias a que integra componentes de hardware y software de Cisco UCS y admite todos los productos Cisco. Su interfaz está hecha en HTML 5 y Java, lo cual permite tener un mejor manejo al usuario. En si este sistema de gestión se basa en modelos que permiten simplificar procesos frecuentes de actualización, monitoreo, almacenamiento, conexiones de red, recopilación de estadísticas, etc.

2.4 Windows PowerShell

PowerShell es una consola de línea de comandos por medio de un lenguaje de scripts. Los cuales se los obtiene gracias al *framework* .NET que literalmente es una biblioteca muy grande de la cual se parte para crear objetos. Con el propósito de automatizar y controlar tareas de forma más efectiva.

2.5 Amazon Web Services

Amazon Web Services es una plataforma en la nube que está formada por un conjunto de servicios remotos. Sus principales servicios son Amazon EC2 y Amazon S3 pero existen más como Lambda y Alexa que junto al servicio S3 son lo que se van a emplear en la realización de este tema de titulación.

2.5.1 Amazon S3

EL servicio de almacenamiento simple (en inglés, *Simple Storage Service o S3*) ofrece escalabilidad, disponibilidad de datos, seguridad y rendimiento. Además, es el encargado del almacenamiento en Internet en donde se guardan datos a modo de objetos, de esta forma se tiene la posibilidad de poder recuperar dichos datos desde cualquier parte de la web.

Amazon S3 se diseñó para ofrecer a los usuarios una durabilidad del 99,999999999 % y puede almacenar millones de datos de diferentes

aplicaciones como es el caso de *Netflix* que es ofrece un servicio de *streaming* a sus subscriptores y su contenido es distribuido desde el servicio de Amazon *S3*. ("AWS | Almacenamiento de datos seguro en la nube (S3)", 2019).

A la hora de hablar del término de durabilidad en el almacenamiento de datos en la nube, a la hora de almacenar una gran cantidad de datos el porcentaje de problemas en el almacenamiento es elevado por lo que una durabilidad alta hace frente a problemas como fallos mecánicos y humanos, desastres naturales entre otros.

En la Figura 1, se puede observar el funcionamiento de las operaciones del S3.

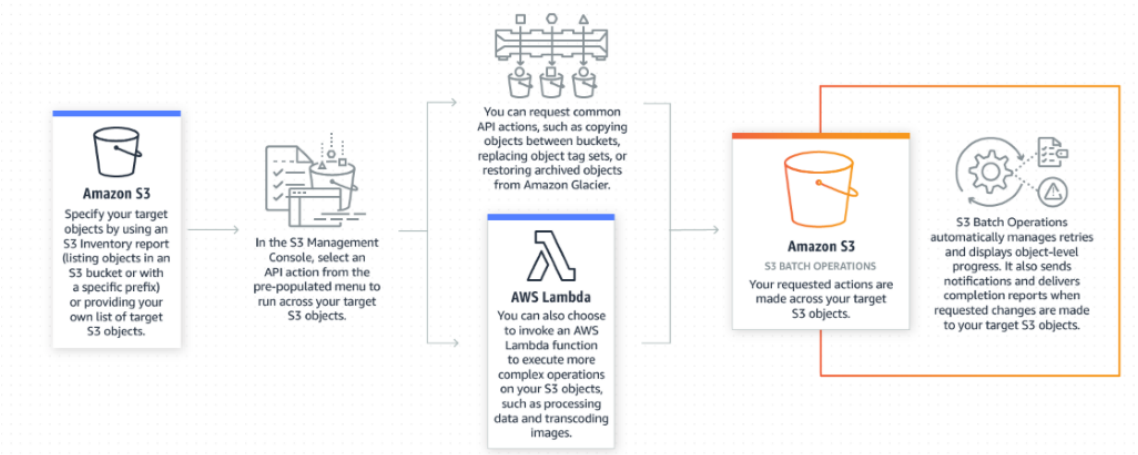


Figura 1. Esquema de funcionamiento del Amazon S3

Adaptado de "Amazon Simple Storage Service: Guía de introducción", 2019.

En primer lugar, Amazon S3 determina sus objetos finales usando su inventario o facilitando su lista de objetos. A continuación, se elige una acción API (copiar, reemplazar o restaurar objetos) en su consola de administración para poder ejecutar los objetos.

Otra posibilidad que tiene *S3* es poder recurrir a una función *AWS Lambda* donde puede realizar operaciones de mayor complejidad en sus objetos (procesamiento de datos y transcodificación de imágenes). Una vez elegidas las acciones, estas se procesan en los objetos *S3* de destino.

Finalmente, las operaciones por lotes del *S3* se encargan automáticamente de la administración de los reintentos, mostrando su respectivo progreso a nivel de objeto con la posibilidad de poder enviar informes y notificaciones cuando se ejecutan cambios en las solicitudes de los objetos *S3* de destino.

2.5.2 Amazon Lambda

"*AWS Lambda* ejecuta el código en una infraestructura informática de alta disponibilidad y ejecuta la administración integral de los recursos informáticos, incluido el mantenimiento del servidor y del sistema operativo, el aprovisionamiento de capacidad y el escalado automático, así como la monitorización y los registros." ("AWS Lambda: Guía para desarrolladores", 2019).

Este tipo de servicio de Amazon tiene la capacidad de ejecutar códigos en todo tipo de aplicaciones o servicios *back-end* sin la necesidad de administrar servidores. Además, una de sus principales características es que este servicio solo ejecuta código cuando es necesario permitiendo configurar dicho código para que su activación se realice de forma automática desde una aplicación (web o móvil) o desde los servicios que ofrece *AWS*. En cuanto a su costo de uso está relacionado con el consumo, es decir, se paga únicamente el tiempo que el código se encuentra en ejecución.

Es de gran importancia tener en cuenta que este servicio de Amazon se puede usar siempre y cuando los códigos de las diferentes aplicaciones permitan la escritura de lenguajes aceptados por *AWS Lambda*.

2.5.3 Alexa Web Information Service (AWIS)

"*AWIS* ofrece una plataforma donde se pueden crear soluciones y servicios web innovadores basados en la gran cantidad de información que tiene Alexa sobre sitios web, accesible con una *API* de servicios web." ("Alexa Web Information Service", 2019).

Otro servicio es Alexa para negocios que tiene herramientas necesarias para gestionar dispositivos compatibles con Alexa, registro de usuarios y asignación de habilidades. Además, permite crear sus propias habilidades de voz mediante la API de Alexa para negocios. ("What Is Alexa for Business? - Alexa for Business", 2019).

2.6 Amazon Echo Dot 3ra Generación

El dispositivo *Echo Dot* de Amazon es un altavoz inteligente el cual realiza la función de un asistente de voz para el usuario. Este asistente de voz es conocido con el nombre de Alexa y mediante el uso de diferentes habilidades o comandos de voz, Alexa puede realizar diversas acciones. Algunas de las funcionalidades más usadas son las siguientes:

- Reproducción de música.
- Ofrece información al usuario (clima, noticias, historia, tráfico, etc.).
- Realizar compras en diferentes tiendas.
- Recordatorios.
- Operaciones matemáticas.
- Llamadas a otros dispositivos Echo Dot, aplicación de Alexa o *Skype*.
- Control de dispositivos inteligentes del hogar.

El dispositivo Echo Dot 3ra generación posee “Wi-Fi doble banda compatible con redes 802.11 a/b/g/n (2.4 GHz y 5 GHz). Bluetooth, con distribución de audio avanzada para transmisión desde tu teléfono móvil al Echo Dot o a un altavoz Bluetooth. Perfil de control remoto de vídeo y audio para control de dispositivos móviles. El control de voz no es compatible con el sistema Mac OS X. Además, requiere de la aplicación Alexa compatible con dispositivos Fire OS, Android, y IOS.” ("Amazon Echo Dot (3ª Generación) - Asistente personal Alexa", 2019).

2.6.1 Amazon Echo

Este dispositivo realiza la funcionalidad del asistente de voz para el usuario. Una vez que dicho dispositivo empieza a funcionar, siempre va a estar escuchado y

se activa cuando detecta el comando de voz “Alexa”, transmite el comando dicho por el usuario al servicio de Amazon Alexa Cloud y finalmente reproduce la respuesta a través del Amazon Echo.

2.6.2 Alexa Cloud

La nube de Alexa es la encargada de manejar el reconocimiento de voz y posteriormente asignar el comando voz correcto. Estos comandos son conocidos como habilidades de Alexa que se encuentran en un servidor y son operados por Amazon. Una vez que la nube de Alexa recibe del servidor de habilidades de Amazon una respuesta, a continuación, lo devuelve al dispositivo de Amazon Echo.

2.6.3 Amazon Skill Server

Las *skills* o habilidades de Amazon son básicamente aplicaciones o tareas que puede realizar Alexa a petición de un usuario y que dichas habilidades se encuentran en los servidores de la nube de Alexa. Además, el usuario puede acceder a todas las habilidades desde la aplicación o desde la página web de Amazon Alexa.

2.6.4 Aplicación de Alexa

La aplicación móvil de Amazon relacionada con la gestión de los servicios Alexa, permite a los usuarios realizar sus propias configuraciones tanto de las diferentes habilidades como de la gestión de los dispositivos Amazon que realicen la función de asistente de voz.

Cuando los usuarios interactúan con sus dispositivos, el servicio *cloud* de Alexa envía informes a la aplicación sobre esas interacciones realizadas. Es de gran importancia saber que la información obtenida viene incluida una grabación de la pregunta realizada por el usuario a su respectivo dispositivo de Amazon Echo.

2.7 Arquitectura de Amazon Alexa

Ofrece un asistente de voz denominado Alexa, el cual se encuentra integrado en los diferentes dispositivos de Amazon Echo y por los cuales los diferentes usuarios acceden al servicio de Amazon Alexa mediante el empleo de comandos de voz. En la figura 2, se puede diferenciar los componentes que conforman su arquitectura.

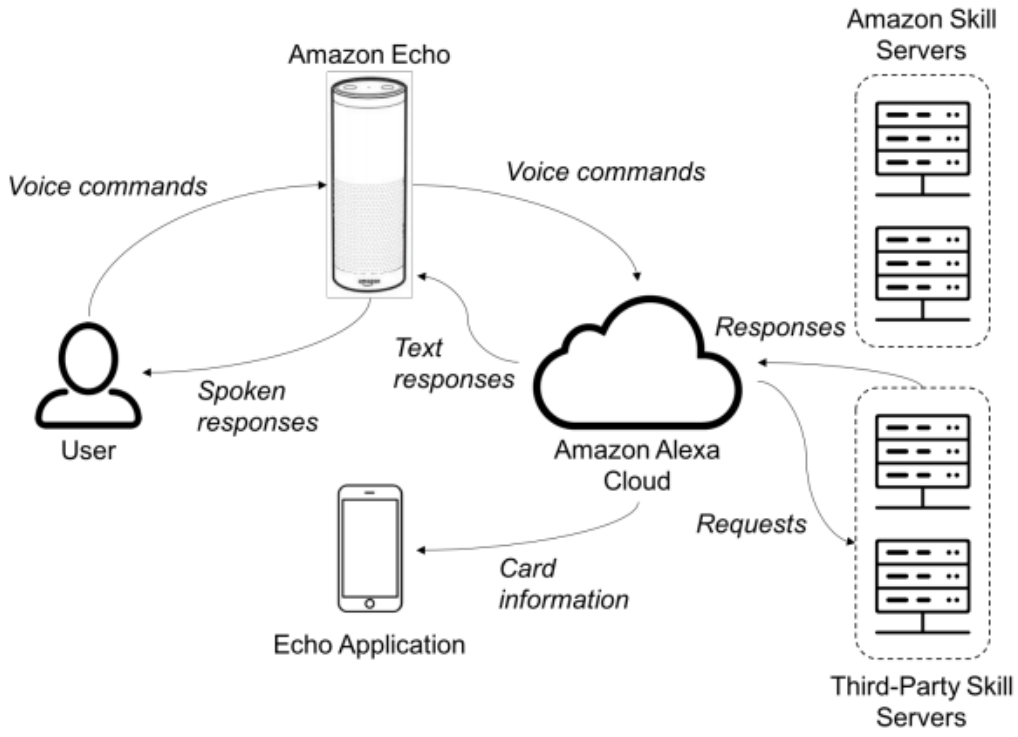


Figura 2. Arquitectura del sistema de Amazon Alexa y Amazon Echo.

Adaptado de “Haack, Severance, Wallace & Wohlwend”, 2017.

3. Capítulo III. Desarrollo e Implementación

3.1 Propuesta de diseño

En el siguiente diseño propuesto, se muestra el funcionamiento de todas las partes que intervienen. Los componentes más importantes son las plataformas de servicio de información web de Alexa (*AWIS*), servicio web de Amazon (*AWS*) y UCS Manager.

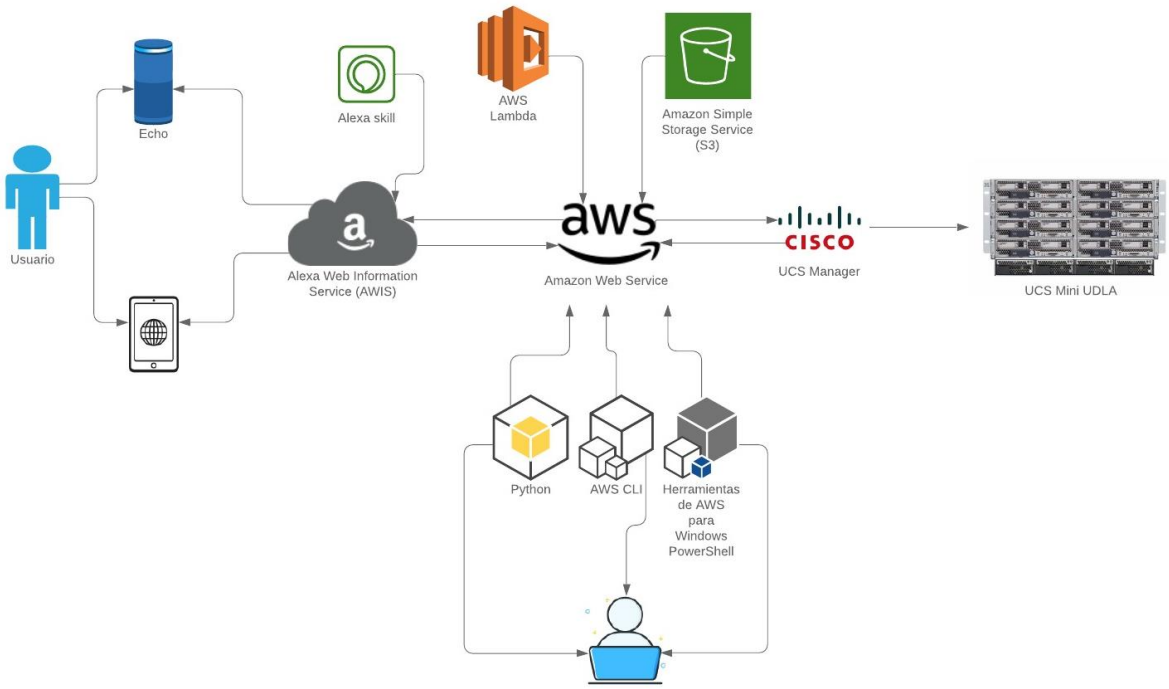


Figura 3. Arquitectura.

Autores.

En la plataforma *AWS* es donde se va a realizar la carga y compilación del código hecho en *Python*, para ello se va a emplear la herramienta de *Windows PowerShell* como intermediario en el cual se efectuará las respectivas configuraciones tanto en la descarga del código como en la instalación de la consola *AWS CLI* para generar automáticamente los servicios. El primero de

ellos es el servicio de almacenamiento simple de Amazon (*S3*) en el cual se almacena el código. El segundo servicio empleado es el *AWS Lambda*, dicho servicio es el que realiza la función del compilador.

En la plataforma *AWIS* se van a generar las diferentes *skills* que ejecutará el prototipo propuesto en este trabajo de titulación. Cabe aclarar que las plataformas *AWS* y *AWIS* se encuentran vinculadas debido a que ambas cuentas le pertenecen al mismo usuario.

UCS Manager es un software con interfaz gráfica en el cual se pueden realizar diferentes configuraciones, en este caso, en el UCS Mini del centro de datos académico de la Universidad de las Américas. La unión del UCS Manager con las plataformas de Amazon se da en la configuración hecha en la herramienta *PowerShell* mencionada anteriormente en la cual se debe introducir la dirección IP del UCS Manager.

Una vez realizado todo correctamente se da paso a los dispositivos finales Amazon Echo y un celular con la aplicación de Alexa instalada. Primero se debe vincular ambos dispositivos y posteriormente ingresar a la aplicación de Alexa y se activa la habilidad que se configuró anteriormente en la plataforma de *AWIS*. Una vez activada, el usuario podrá iniciar dicha habilidad mediante el asistente de voz Echo y realizar las configuraciones directamente en el UCS Manager mediante la voz.

3.2 Creación y Configuración plataformas de Amazon

3.2.1 Amazon Web Server

Para utilizar esta plataforma es necesario crear un usuario, el cual se lo hace en el siguiente enlace: https://aws.amazon.com/es/. Para crear un usuario es necesario dar clic derecho en la parte donde dice inicie sesión en la consola como lo muestra en la figura 4. Posteriormente se despliega la siguiente pantalla como se observa la figura 5.

Figura 4. Sitio oficial de *AWS*.

,

Figura 5. Crear o iniciar sesión en *AWS*.

Dar clic en la opción Crear una cuenta en *AWS.* Se despliega la pantalla de la figura 6, y será necesario ingresar todos los datos solicitados.

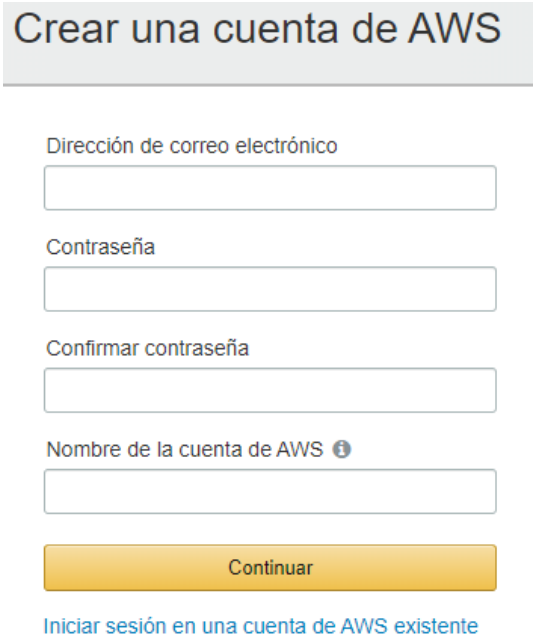


Figura 6. Creación de la cuenta en *AWS*.

Una vez creada la cuenta, si es de tipo gratuito, la activación se demora 24 horas. Cuando la cuenta se encuentre disponible se tendrá acceso a todos los servicios que esta plataforma brinda, y en específico para este proyecto se utilizará *Lambda*, *S3*.

Figura 7. Consola administración en *AWS*.

3.2.2 Amazon Alexa

Al igual que en la plataforma *AWS* es necesario crear un usuario, por lo que se debe ingresar en el siguiente link: https://developer.amazon.com/en-US/alexa. Donde se procede a dar clic en la opción de Registrarse que se encuentra en la parte superior derecha como se indica en la Figura 8.

Figura 8. Sitio oficial Amazon Alexa.

En la figura 9 que se puede observar a continuación, es necesario dar clic en la opción Crea tu cuenta con Amazon.

No es la misma cuenta que se crea en *AWS* ya que en este caso es solo para desarrollar las habilidades de Amazon Alexa, en la otra plataforma es para utilizar los servicios de Amazon.

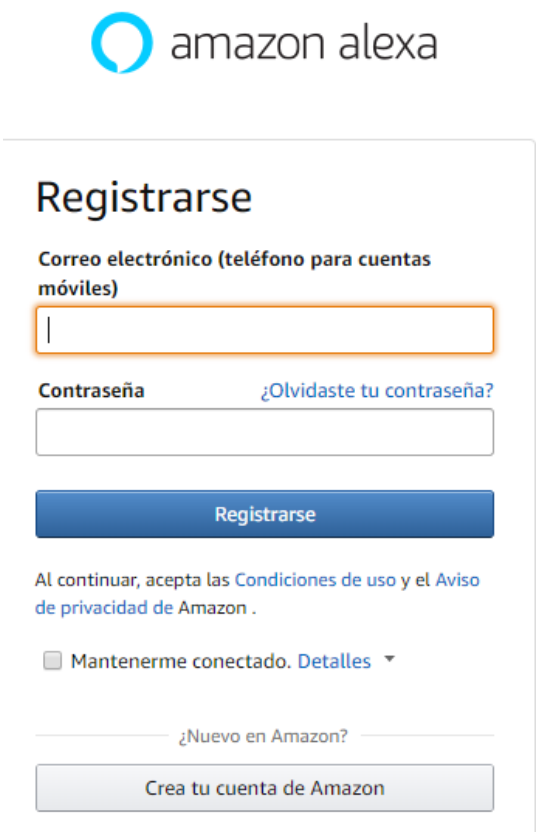


Figura 9. Crear o iniciar sesión en Amazon Alexa.

Se despliega la pantalla de la figura 10, y es necesario ingresar todos los datos solicitados.

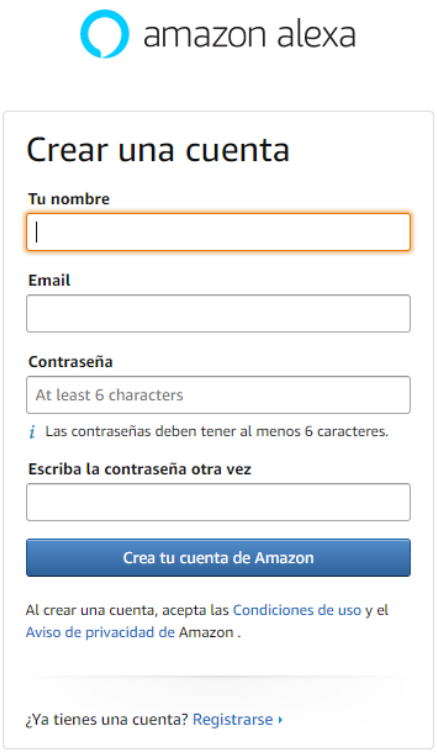


Figura 10. Crear usuario en Amazon Alexa

Cuando se tiene ya el usuario creado, en la parte superior izquierda se da clic en la opción de *Alexa Developer Console*. Donde se crea las habilidades para vincular al Amazon Echo mediante la aplicación de Amazon Alexa.

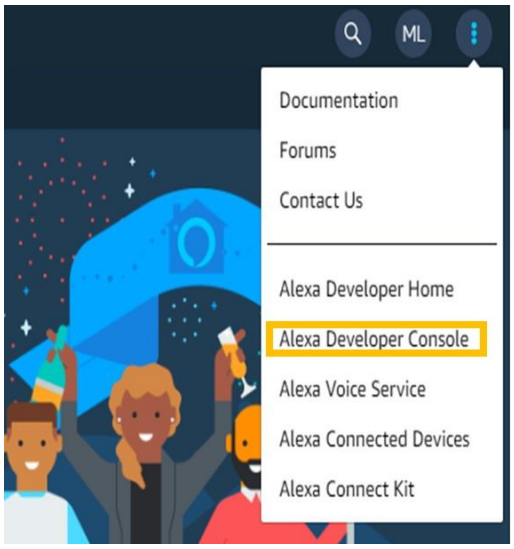


Figura 11. Consola para desarrollar las habilidades de Alexa.

3.3 Configuración en PowerShell

En esta parte se detallarán paso a paso las configuraciones realizas en la plataforma de *PowerShell.* Se debe crear un directorio con el nombre de “Alexa” y se posteriormente se ingresará en él.

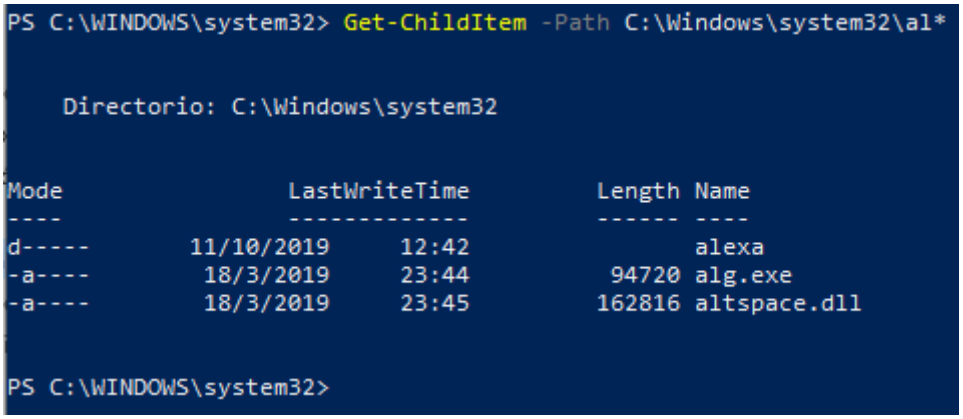


Figura 12. Creación y búsqueda del directorio Alexa.

A continuación, se debe obtener los códigos desarrollados en *Python*. Para ello se utilizará el comando: `git clone`, seguido de la dirección en la que se encuentra dicho código. En este caso se encontrarán en la plataforma de *GitHub*.

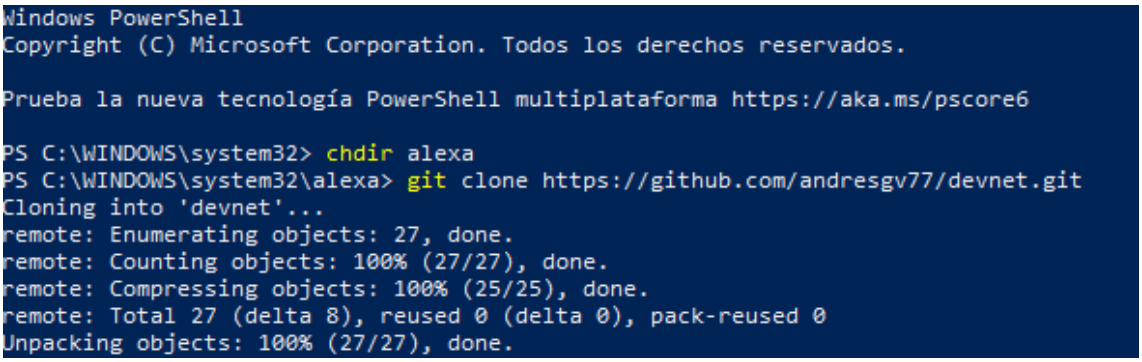


Figura 13. Creación y búsqueda del directorio Alexa.

Una vez realizada la clonación. Se realiza la creación de un *virtual environment* empleando el siguiente comando:

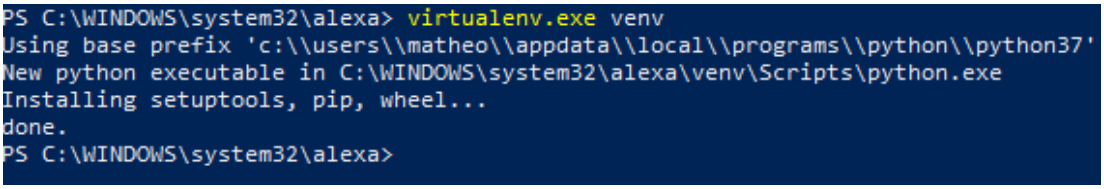


Figura 14. Creación y búsqueda del directorio Alexa.

Es necesario saber que, al emplear el comando de la figura 14 se crea de forma automática un nuevo directorio con el nombre de *venv*.

Se debe realizar una copia del directorio "*devnet*" en el nuevo directorio creado anteriormente, es decir, en el directorio "*venv*". Se ingresa en el directorio "*venv*" y se realiza la activación del *virtual environment* mediante el uso del siguiente comando:

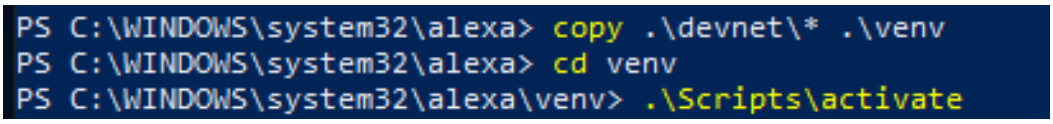


Figura 15. Comando:.`\Scripts\activate`

Como se observa en la figura 15, el nombre sufrió un cambio. Ahora se debe instalar el *AWS CLI,* la cual es una herramienta que permite administrar varios servicios de Amazon por medio de línea de comandos. Para ello se emplea el siguiente comando:

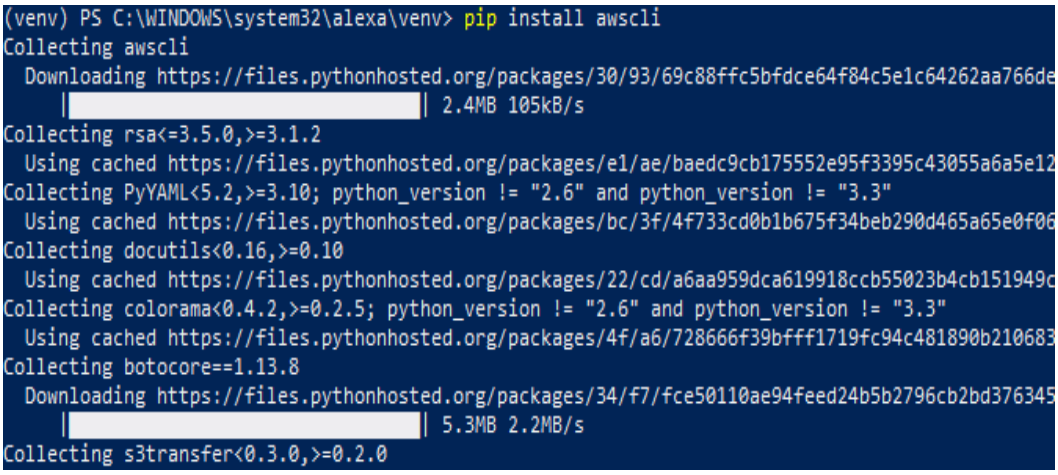


Figura 16. Comando: `pip install awscli.`

Para ingresar la cuenta creada en el paso 3.2.1 se utiliza el comando: *AWS configure*. Donde lo primero que pide es el ID que se obtiene en la configuración de cuenta, al igual que la *Secret ID Key*, como no existe una región para América del Sur se utiliza la región de Virginia y el formato es *JSON.*

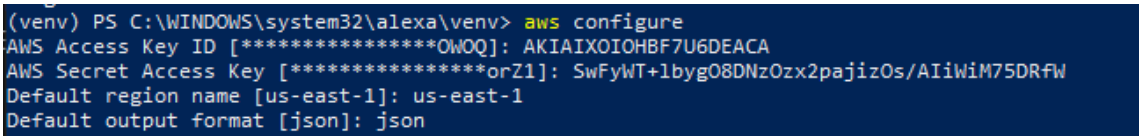


Figura 17. Comando: `AWS configure.`

Con el comando: `.\zip.exe` se comprime todo en un archivo zip para que se pueda cargar en el S3 del Amazon Web Service.

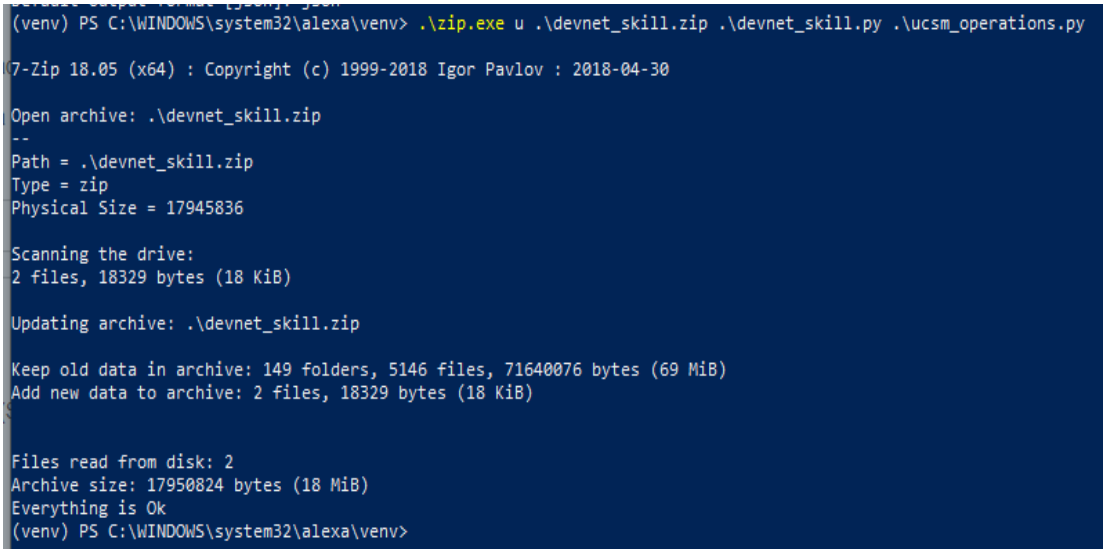


Figura 18. Comando: `.\zip.exe`

Para crear un espacio de almacenamiento en el servicio *S3* en el servicio web de Amazon se lo hace por medio del comando:

```
aws s3 mb s3://ruta
```

Ya con un lugar donde almacenar en *S3* por medio del comando: `aws s3 cp`, se copia lo que se encuentra en el directorio en el depósito de S3 creado anteriormente.

Para verificar que se haya creado correctamente el espacio de almacenamiento y en el mismo se creó la copia se lo hace mediante el comando: `aws s3 ls s3://ruta`.

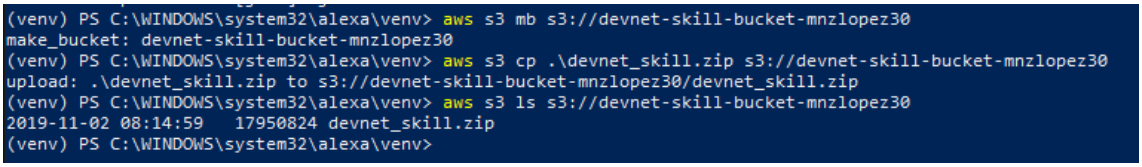


Figura 19. Comandos: `aws s3 mb s3://ruta`

`aws s3 cp .\ruta`

`aws s3 ls s3://ruta`

Cuando ya se encuentra el código en el servicio *S3* es necesario crear en el servicio Lambda mediante el cual se conectará al código almacenado en el servicio *S3*. Al crear esta función Lambda es importante tomar en cuenta los siguientes parámetros que es necesario cambiar:

- --*environment* Variables={UCSHOST=0.0.0.0}: IP publica que tiene el UCS Manager. La cual se toma según la región como se puede observar en el anexo 1.
- --*function-name*: nombre de la función entre comillas “nombre” y posterior de los signos de admiración.
- --*code S3Bucket*: colocar el mismo nombre dado a la función es decir |--code S3Bucket="devnet-skill-bucket-mnzlopez30"|.

Figura 20. Parámetros para configurar al momento de crear la función Lambda.

Para terminar con la creación de la función es necesario iniciar sesión en *Amazon Web Service* y buscar el servicio *Lambda*. Como se observa en la siguiente figura:

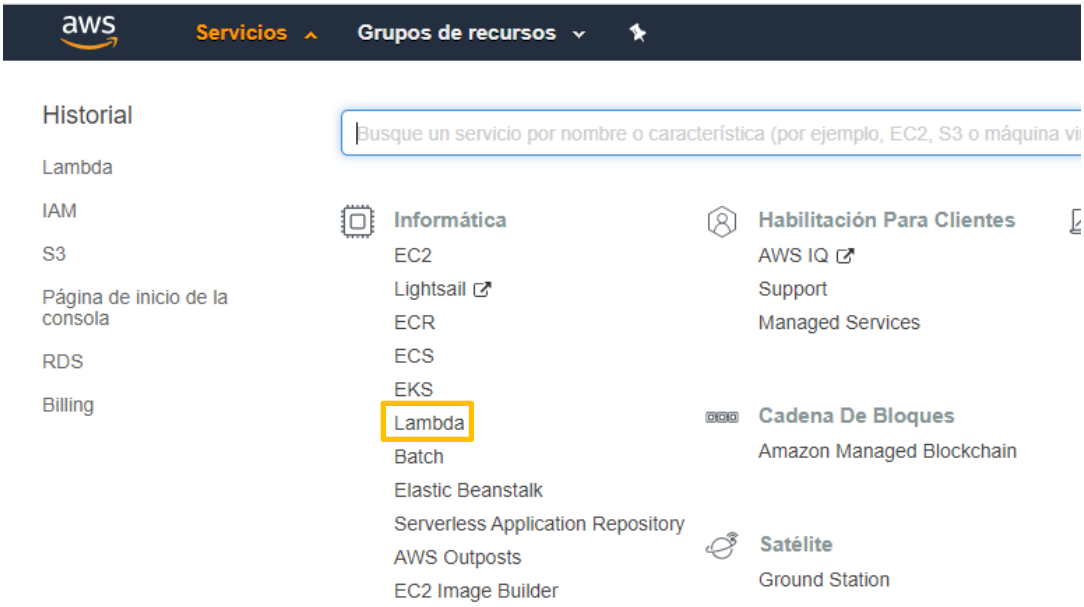


Figura 21. Servicio Lambda en Amazon Web Service.

Se podrá visualizar la función creada por medio de comandos en *PowerShell* y donde es necesario seleccionar y dar clic derecho para ingresar a la función para configurar los roles correspondientes.

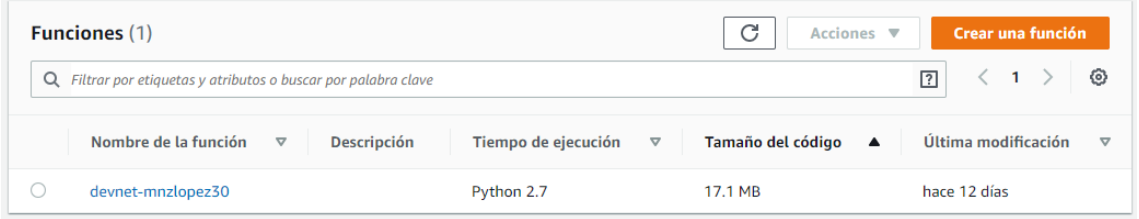


Figura 22. Función creada en el servicio Lambda.

Se debe añadir un *trigger* de *Alexa Skills Kit* y se debe deshabilitar la verificación del ID de la habilidad se va a crear posteriormente en la plataforma de Amazon Alexa.

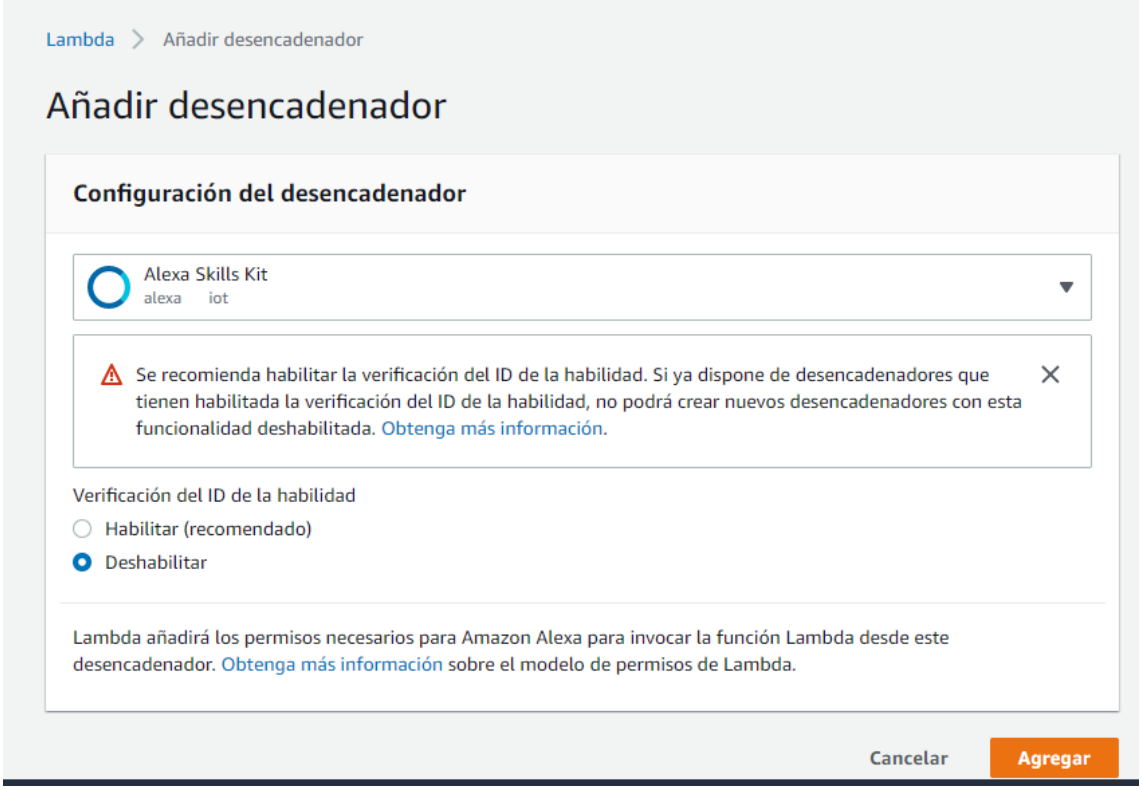


Figura 23. Configuración y creación del *trigger Alexa Skills Kit.*

Una vez que se agrega el *trigger* necesario, hay que crear un rol para el servicio *S3* donde se encuentra almacenado el código. Dentro de la función creada en *Lambda* existe la opción de crear o hacer uso de un rol ya creado basado en el servicio *S3*.

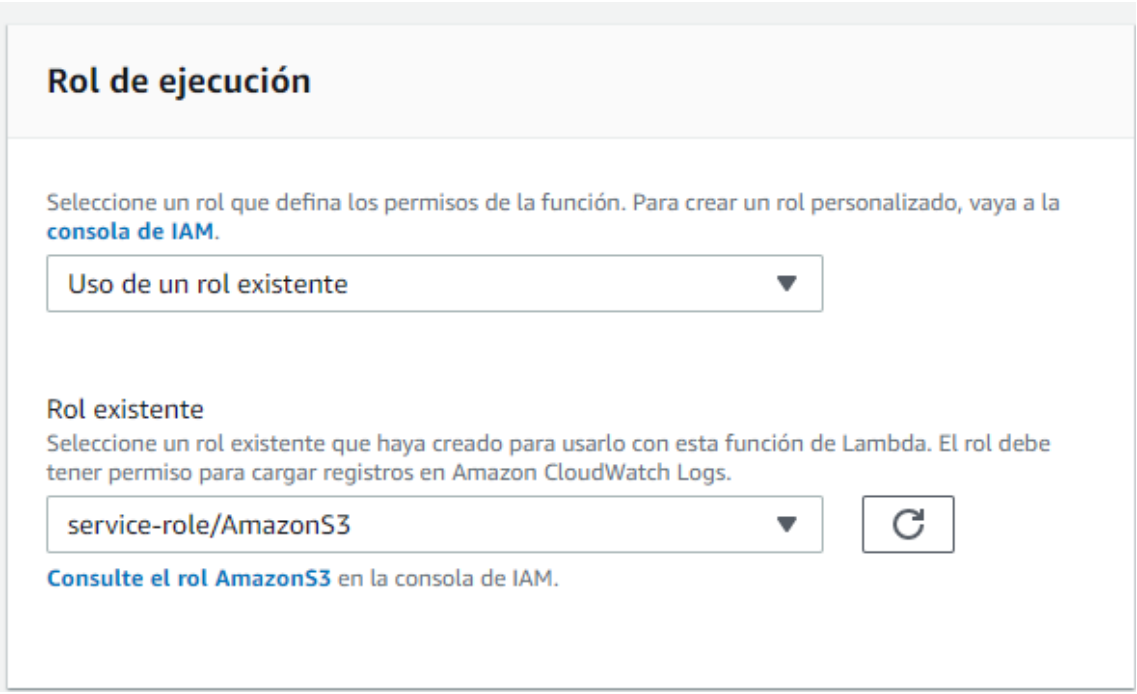


Figura 24. Crear un rol para S3.

Cuando se crea el rol y se agrega el *trigger* el diseño cambia como se puede observar en la figura siguiente, pero en ocasiones si el gráfico no es igual o tiene diferentes diseños, hay que verificar que el rol se encuentre bien creado o que sea el adecuado.

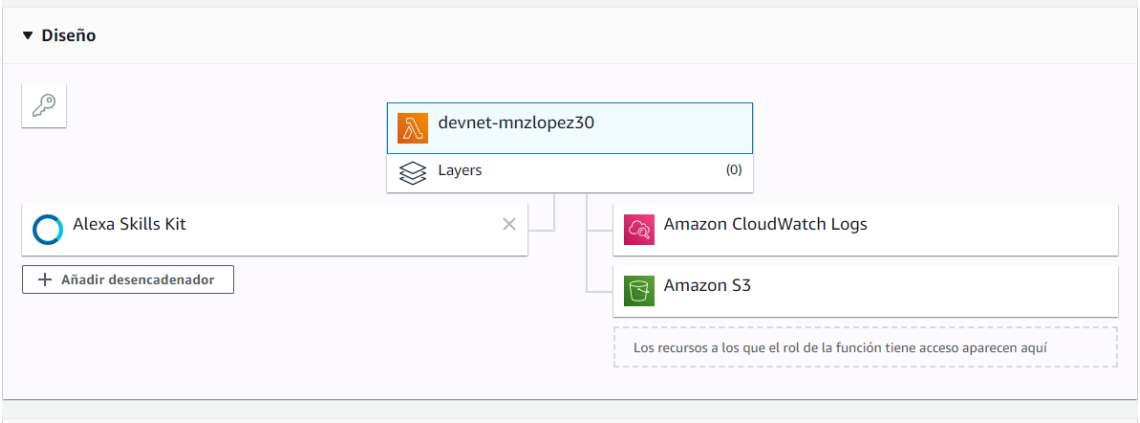


Figura 25. Diseño en la función *Lambda*.

Si el diseño no es igual que el que se encuentra en la figura anterior no necesariamente está mal, se debe verificar en la parte de permisos que estén el del servicio *S3*.

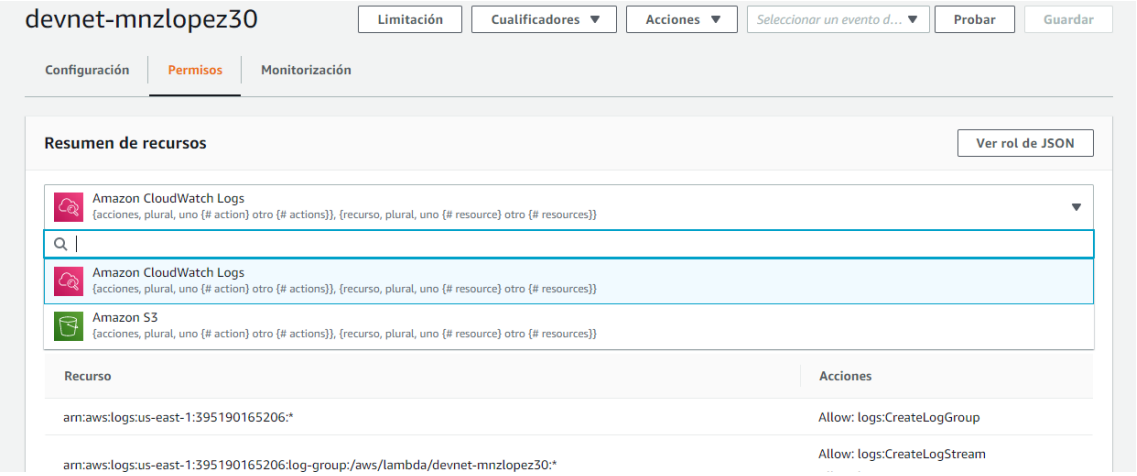


Figura 26. Permisos agregados a la función creada.

3.4 Creación de la habilidad en Amazon Alexa

En este apartado se detallará la creación de las diferentes *skills* o habilidades propuestas en la plataforma de Amazon Alexa.

Se debe ingresar a la consola de Amazon Alexa, para ello se escoge la opción *Alexa Developer Console* como se puede observar en la Figura 27.

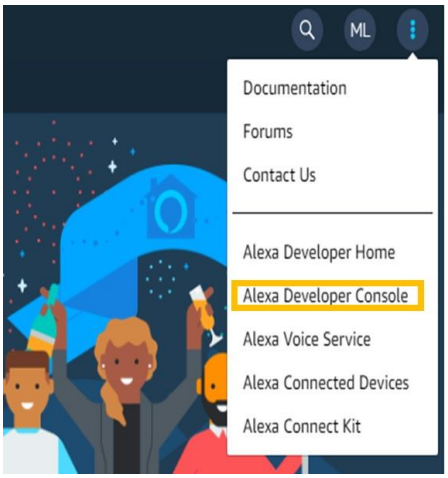


Figura 27. Ingreso a la consola de Amazon Alexa.

A continuación, se debe crear una habilidad como se observar en la Figura 28.

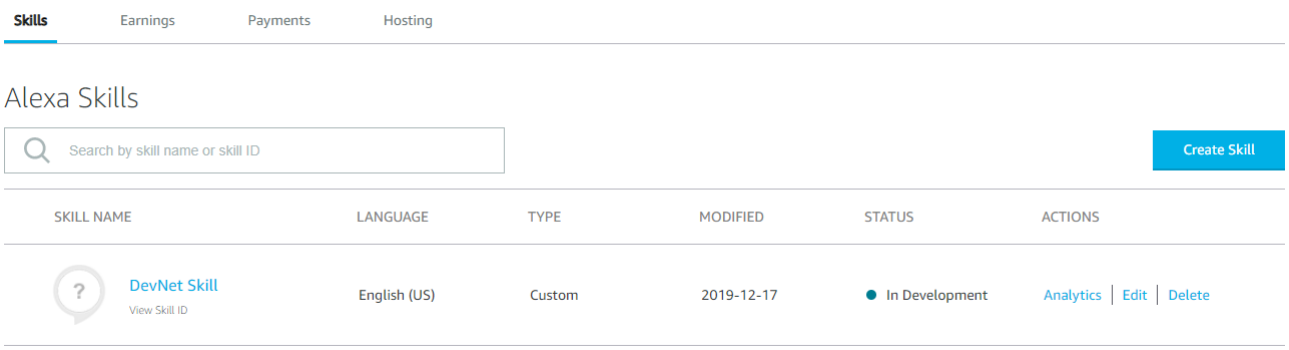


Figura 28. Creación de la habilidad.

Dentro de dicha habilidad se van a configurar las diferentes intenciones, pero previamente hay que establecer el nombre de la invocación de la *habilidad*, es decir, será la palabra clave para poder iniciar la habilidad. En este caso se usó la palabra *devnet*.

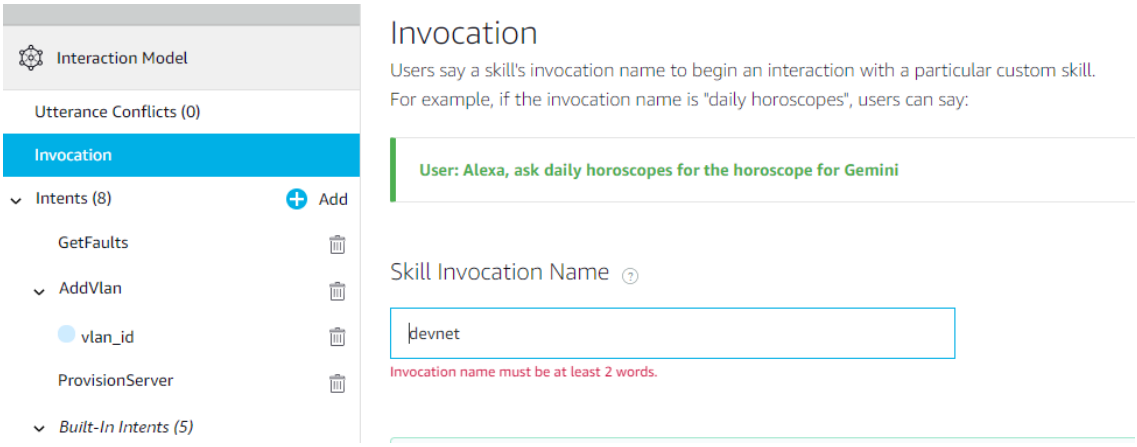


Figura 29. Configuración del nombre de invocación.

La primera intención propuesta en la habilidad de Alexa es la de "*GetFaults*" la cual se activa mediante la frase "*What is my fault count*". El dispositivo Amazon Echo, recibe la petición por parte del usuario, en este caso es la frase mencionada anteriormente, a continuación, es activada la intención "*GetFaults*" y se empieza a procesar internamente el código *Python* y devuelve la petición a

través del asistente de voz. El principal objetivo de esta intención es realizar la cuenta de fallos que tiene el UCS Manager, ya sean fallos críticos, menores, mayores y advertencias.

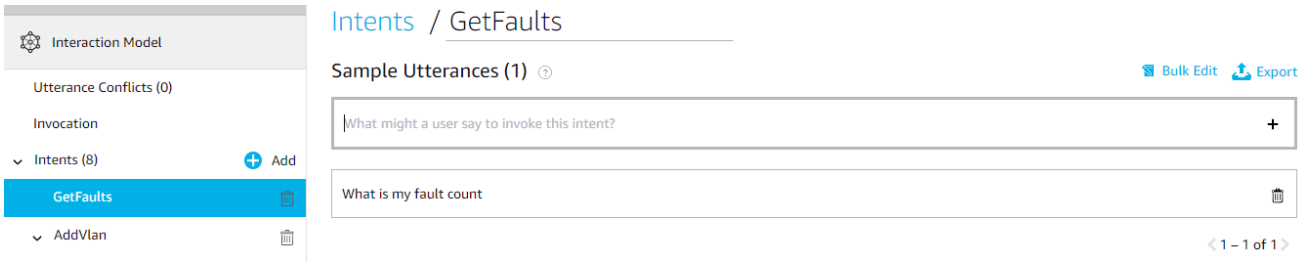


Figura 30. Configuración de la intención GetFaults.

La segunda intención propuesta es la de "*AddVlan*". Dicha intención es iniciada mediante la frase "*Add V Lan {vlan_id}*" en donde la parte del "*vlan_id*" se refiere al número que el usuario desea dar a la hora de crear una *VLAN*. Como se mencionó anteriormente, una vez iniciada la intención se empieza a procesar internamente el código *Python* y devuelve la petición a través del asistente de voz Echo de Amazon. Es de gran importancia mencionar que el valor seleccionado para la "*vlan_id*" es Amazon Number ya que es un variable numérica como se puede observar en la Figura 31.

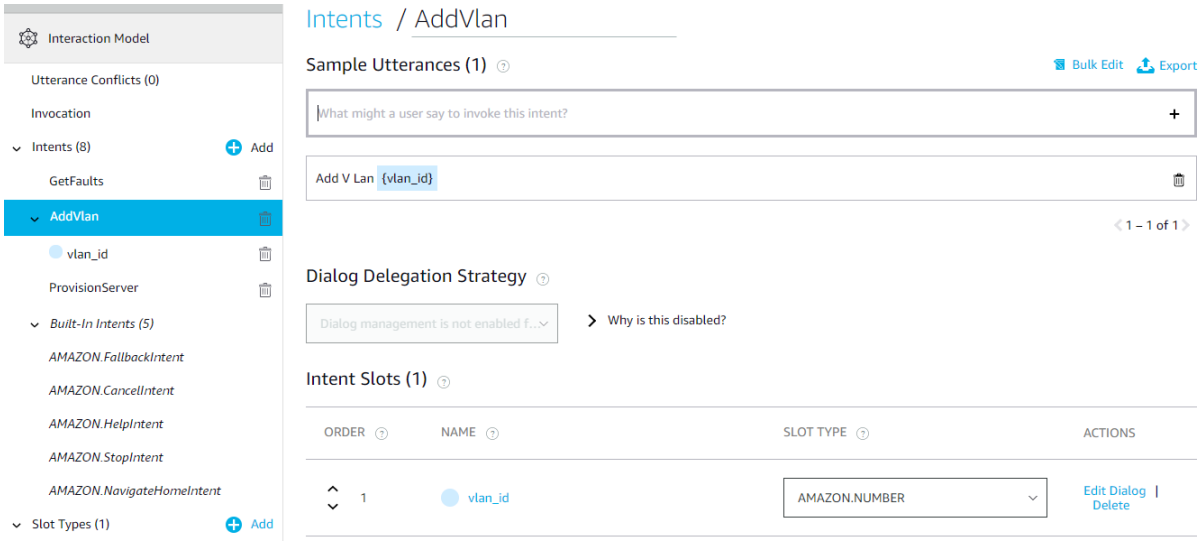


Figura 31. Configuración de la intención *AddVlan*.

La tercera intención propuesta es la de "*RemoveVlan*". Esta intención es iniciada con la frase "*Remove V Lan {vlan_id}*" en donde la parte del "*vlan_id*" pasa lo mismo que se comentó en la segunda intención de *AddVlan*.

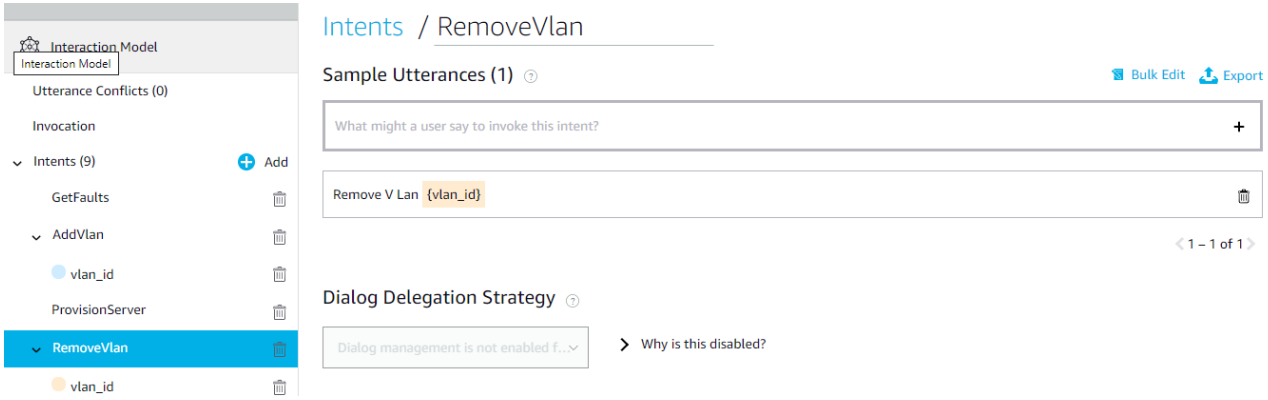


Figura 32. Configuración de la intención RemoveVlan.

La cuarta intención propuesta es la de "*ProvisionServer*". La frase que activa esta intención es "*Provision a server*". Su principal objetivo es realizar el aprovisionamiento de un servidor dentro del UCS Manager. una vez iniciada la intención se empieza a procesar internamente el código *Python* y devuelve la petición a través del asistente de voz Echo de Amazon.

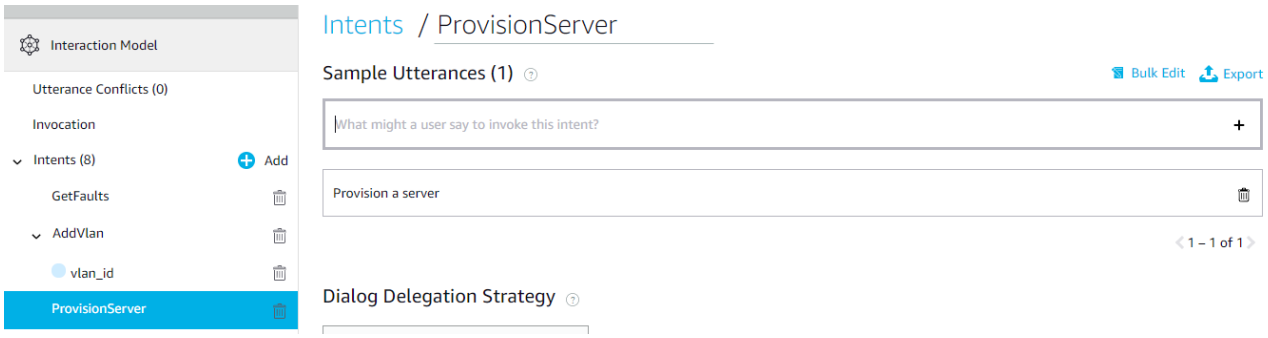


Figura 33. Configuración de la intención ProvisionServer.

Una vez creadas todas las intenciones dentro de la habilidad, se procede a construir y guardar el modelo como se puede observar en la siguiente Figura 34.

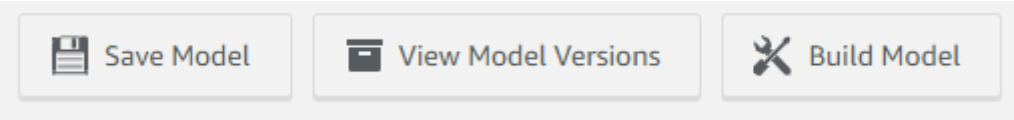


Figura 34. Construcción y guardado del modelo.

Por último, se debe ir a la opción de "*EndPonit*" donde se efectuará la conexión de la habilidad de Alexa previamente configurada con la función Lambda. Una vez dentro de dicha opción, se debe escoger la opción "*AWS Lambda ARN*" y se copia y se pega la dirección *ARN* en donde se muestra en la Figura 35. Finalmente, se debe guardar en la opción "*Save Endponits*".

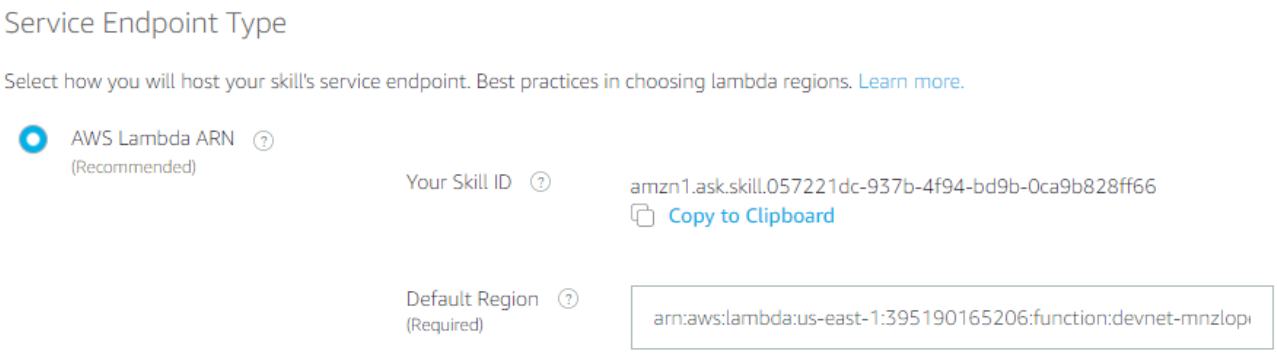


Figura 35. Configuración del EndPonit.

3.5 Creación de código en *Python*

Existen dos clases principales en el programa. La primera clase se llama ucsm_operations.py en la cual se puede encontrar las líneas del código para la creación y eliminación de una *VLAN*, creación de un *service profile* y como obtener información específica del UCS Manager.

Un método sumamente importante es el de *login* ya que se ingresan las credenciales del UCS Manager como se observa en el recuadro amarillo de la Figura 36 donde se específica el usuario y la contraseña.

```
def ucsm_login():

    global handle, status

    username = "admin"
    password = "[redacted]"

    try:
        handle = UcsHandle(ucsmhost,username,password)
        if handle.login() == True:
            status['login'] = "success"
            return

    except urllib2.URLError as err:
        status['login'] = "URLError"
        return

    except UcsException as err:
        status['login'] = "UcsException"
        return


def ucsm_logout():

    handle.logout()
```

Figura 36. Credenciales del UCS Manager.

En la siguiente Figura 37 se encuentra el método para añadir una *VLAN*.

```
# Add a VLAN to UCS Manager
def add_ucs_vlan(vlan_id):

    vlan_id_num = int(vlan_id)

    from ucsmsdk.mometa.fabric.FabricVlan import FabricVlan

    if vlan_id == "1":
        message = ("For the requested UCS Manager V Lan add operation, " +
            "V Lan 1 can be given additional names, however, this skill does not allow for that procedure.")
        return message
    elif ((vlan_id_num <= 1) or
        (vlan_id_num >= 4030 and vlan_id_num <= 4048) or
        (vlan_id_num > 4093)):
        message = ("For the requested UCS Manager V Lan add operation, " +
            "the provided V Lan I D " + vlan_id + ", is not allowed.")
        return message

    ucsm_login()
```

Figura 37. Método añadir *VLAN*.

En donde se crea una variable de tipo entero y es necesario invocar las librerías necesarias. Se utiliza una sentencia repetitiva en este caso un *if else* donde no se puede modificar, ni borrar la *VLAN* 1 dando un mensaje en caso de querer realizar cualquier configuración con la *VLAN* 1.

Se puede añadir una *VLAN* entre un rango especifico, si ya existe da un mensaje de que ya se encuentra en el UCS Manager, caso contrario crea la *VLAN* como se puede observar en la Figura 38:

```
            response = handle.query_dn("fabric/lan/net-vlan" + vlan_id)

            if response and response.name == "vlan" + vlan_id:
                message = "V Lan " + vlan_id + " has been added to UCS Manager."
            else:
                message = "V Lan " + vlan_id + " was not added to UCS Manager."
        else:
            message = "V Lan " + vlan_id + " already exists on UCS Manager."
```

Figura 38. Mensajes cuando se dan algunos casos específicos al momento de crear una *VLAN*.

Posteriormente invoca al método *login* donde busca las credenciales para ingresar al UCS Manager. Una vez que se ingresa con las credenciales correspondientes verifica si la IP que se encuentra almacenada en el servicio *S3* de Amazon tiene conexión, caso contrario da un mensaje de que no puede conectarse con el UCS Manager.

Nota: Tiene que ser una IP Pública de lo contrario no será posible establecer una conexión. La cual se toma según la región como se puede observar en el anexo 1.

```
if status['login'] == "success":
    response = handle.query_dn("fabric/lan/net-vlan" + vlan_id)
else:
    message = ("there was an error connecting to the UCS Manager, " +
        "please check the access credentials or IP address")
    return message

if not response:
    fabric_lan_cloud = handle.query_dn("fabric/lan")
    vlan = FabricVlan(parent_mo_or_dn=fabric_lan_cloud,
                      name="vlan" + vlan_id,
                      id=vlan_id)

    handle.add_mo(vlan)
    handle.commit()

    response = handle.query_dn("fabric/lan/net-vlan" + vlan_id)
```

Figura 39. Método añadir *VLAN*

El siguiente método es para ver los fallos que se encuentra en el UCS Manager. Primero ingresa al UCS Manager con las credenciales que se encuentran en el método de *login*.

Se crea variables con los nombres de los fallos iguales que tiene el UCS Manager es decir *crítico, mayor, menor y advertencia*. Por medio de una sentencia de control se va ingresando un contador al que se agrega el número y

el tipo de fallos, posteriormente Alexa proporcionará un resumen al administrador de red.

Con un mensaje respectivo nos da un total de cada fallo que tiene en ese momento el UCS Manager ya sean fallos críticos, menores, mayores y advertencias como se puede observar en la Figura 40.

```
# Retrieve fault counts for, critical, major, minor and warning faults from UCS Manager
def get_ucs_faults():

    ucsm_login()

    if status['login'] == "success":
        response = handle.query_classid("FaultInst")
    else:
        message = ("there was an error connecting to the UCS Manager, " +
            "please check the access credentials or IP address")
        return message

    ucsm_logout()

    sev_critical = 0
    sev_major    = 0
    sev_minor    = 0
    sev_warning  = 0


for fault in response:
    if fault.severity == 'critical':
        sev_critical += 1
    elif fault.severity == 'major':
        sev_major += 1
    elif fault.severity == 'minor':
        sev_minor += 1
    elif fault.severity == 'warning':
        sev_warning += 1

message = ("For the requested UCS Manager fault retrieval operation, there are " +
    str(sev_critical) + " critical faults, " +
    str(sev_major) + " major faults, " +
    str(sev_minor) + " minor faults, and " +
    str(sev_warning) + " warnings")

return message
```

Figura 40. Método contador de fallos.

Otro metodo empleado es el de crear un *service profile* que para su correcto funcionamiento es necesario importar algunas librerias necesarias.

Al igual que todos los metodos es necesario que busque las credenciales en el método *login* para poder ingresar al UCS Manager. Se crea una organización en root que esta dentro del *service profile*, donde se define el nombre que se quiera ingresar en este caso como en la Figura 41. Para las suborganizaciones tengan el mismo nombre de la organización y se vayan creando secuencialmente.

Se ingresan los parametros necesarios para que nos de la informacion del chasis y del servidor.

```
# Create and Associate a Service Profile to an available server
def set_ucs_server():

    from ucsmsdk.mometa.ls.LsServer import LsServer
    from ucsmsdk.mometa.ls.LsBinding import LsBinding
    from ucsmsdk.mometa.macpool.MacpoolPool import MacpoolPool
    from ucsmsdk.mometa.macpool.MacpoolBlock import MacpoolBlock
    from ucsmsdk.mometa.org.OrgOrg import OrgOrg

    ucsm_login()

    if status['login'] != "success":
        message = ("there was an error connecting to the UCS Manager, " +
            "please check the access credentials or IP address")
        return message

    # Create Org DevNet
    mo_org = OrgOrg(parent_mo_or_dn="org-root", name="DevNet")
    handle.add_mo(mo_org, modify_present=True)
    handle.commit()
```

```
# Create the Service Profile
mo_sp = LsServer(parent_mo_or_dn="org-root/org-DevNet", src_templ_name="DevNet_Skill_Template", name=service_profile_name)
mo_sp_templ_ls_binding = LsBinding(parent_mo_or_dn=mo_sp, pn_dn=blade.dn)
handle.add_mo(mo_sp, modify_present=True)
handle.commit()

ucsm_logout()

message = ("For the requested UCS Manager Server Provisioning operation," +
    " server, " + blade.slot_id + ", " +
    " in chassis, " + blade.chassis_id + ", " +
    " has been provisioned with service profile, " + service_profile_name.replace('_',' ') + "," +
    " in the Dev Net Organization.")

return message
```

Figura 41. Método crear un Service Profile.

El siguiente método creado en esta clase consiste en eliminar una *VLAN* del UCS Manager y se usan los mismos parámetros del método de añadir *VLAN*.

Se debe tomar en cuenta que la *VLAN* 1 no es posible eliminar y modificar. Si se intenta hacer eso Alexa nos va a decir un mensaje indicando que no es posible realizar dicha operación con la *VLAN* 1.

En base a una variable del tipo entero busca y elimina según el ID de la *VLAN* y da un mensaje si se eliminó correctamente. Si el usuario intenta eliminar una *VLAN* que no existe en el UC Manager no tendrá ningún efecto.

```
68  # Remove a VLAN from UCS Manager
69  def remove_ucs_vlan(vlan_id):
70
71      if vlan_id == "1":
72          message = ("For the requested UCS Manager V Lan remove operation, " +
73              "this skill does not support the removal of V Lan 1.")
74          return message
75
76      ucsm_login()
77
78      if status['login'] == "success":
79          response = handle.query_dn("fabric/lan/net-vlan" + vlan_id)
80      else:
81          message = ("there was an error connecting to the UCS Manager, " +
82              "please check the access credentials or IP address")
83          return message
84
85      if response and response.name == "vlan" + vlan_id:
86
87          handle.remove_mo(response)
88          handle.commit()
89
90          response = handle.query_dn("fabric/lan/net-vlan" + vlan_id)
91
92           if not response:
93               message = "V Lan " + vlan_id + " has been removed from UCS Manager."
94           else:
95               message = "V Lan " + vlan_id + " was not removed from UCS Manager."
96       else:
97           message = "V Lan " + vlan_id + " does not exist on UCS Manager."
98
99       ucsm_logout()
100
101      return "For the requested UCS Manager V Lan remove operation, " + message
```

Figura 42. Método *remove VLAN.*

La segunda clase se llama devnet_skill.py en donde se encuentra las líneas de código de cómo crear y eliminar una *VLAN*, como crear un *service profile* y como obtener información específica del UCS Manager. La diferencia respecto a la primera clase es que aquí los métodos son invocados de la clase ucsm_operations.py.

En esta función es lo que el Amazon Echo nos va a decir por medio de Alexa, es decir el mensaje de bienvenida donde nos informará el menú que se encuentra en el código y es posible recrear en el UCS Manager.

```
def get_welcome_response():

    session_attributes = {}
    card_title = "Welcome"
    speech_output = "Welcome to the DevNet Alexa Skill for UCS Managment." \
                    "Hi Andres and Matheo." \
                    "You can say things like, What is my fault count?" \
                    "Or you can say Add V Lan 200." \
                    "Or you can say Provision a server."
    # If the user either does not reply to the welcome message or says something
    # that is not understood, they will be prompted again with this text.
    reprompt_text = "Did you want to do something with, or know something about your UCS System?"
    should_end_session = False
    return build_response(session_attributes, build_speechlet_response(
        card_title, speech_output, reprompt_text, should_end_session))
```

Figura 43. Método menú de inicio.

En los siguientes métodos se pretende obtener todos los fallos, crear una *VLAN* y un *service profile* llamando a la función correspondiente que se creó en la clase ucsm_operations.py

```
# Get the UCS Faults
def get_faults(intent, session):
    session_attributes = {}
    reprompt_text = None

    speech_output = ucsm_operations.get_ucs_faults()
    should_end_session = True

    return build_response(session_attributes, build_speechlet_response(
        intent['name'], speech_output, reprompt_text, should_end_session))
```

Figura 44. Método obtener fallos en la clase devnet_skill.py

```
# Add a UCS VLAN
def add_vlan(intent, session):
    session_attributes = {}
    reprompt_text = None

    speech_output = ucsm_operations.add_ucs_vlan(intent['slots']['vlan_id']['value'])
    should_end_session = True

    return build_response(session_attributes, build_speechlet_response(
        intent['name'], speech_output, reprompt_text, should_end_session))
```

Figura 45. Método agregar *VLAN* en la clase devnet_skill.py.

4. Capítulo IV. Pruebas y Resultados

En este punto se van a desarrollar las diferentes pruebas realizadas junto a los resultados obtenidos. Básicamente estas pruebas consisten en probar las diferentes intenciones configuradas dentro de la habilidad de Alexa.

La primera prueba realizada fue la contabilización de fallas del UCS Manager. Primero el usuario activa la *skill* o habilidad diciendo la siguiente frase "*Alexa, ask devnet*" al dispositivo Alexa Echo, esto se debe realizar siempre que se pretenda activar una intención. A continuación, la habilidad empieza a funcionar y dicho dispositivo empieza a decirle el menú de inicio al usuario. Dentro de este menú se encuentran las diferentes intenciones que se configuraron dentro de la habilidad *devnet*.

Una vez que el asistente de voz Alexa termina de decir el menú, el usuario debe pronunciar una de las frases que activan las diferentes intenciones. Para la realización de esta prueba se debe decir la frase "What is my fault count", como consecuencia el asistente de voz devuelve el resultado diciendo el número de fallos críticos (0), mayores (17), menores (1) y advertencias (7).

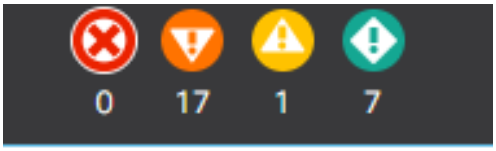


Figura 46. Números de fallos del UCS Manager.

La siguiente prueba realizada fue la creación de *VLANs*. Primero el usuario debe entrar al menú de inicio activando la habilidad mediante la frase "*Alexa, ask devnet*". A continuación, el usuario escucha dicho menú e inicia una intención, que en este caso es la de creación de una *VLAN*. Para ello se debe decir la frase "*Add V Lan*" seguida del número que se quiere asignar a dicha *VLAN*.

En las pruebas realizadas se agregó las *VLANs* 100 y 4049 como se muestran en la Figura 47.

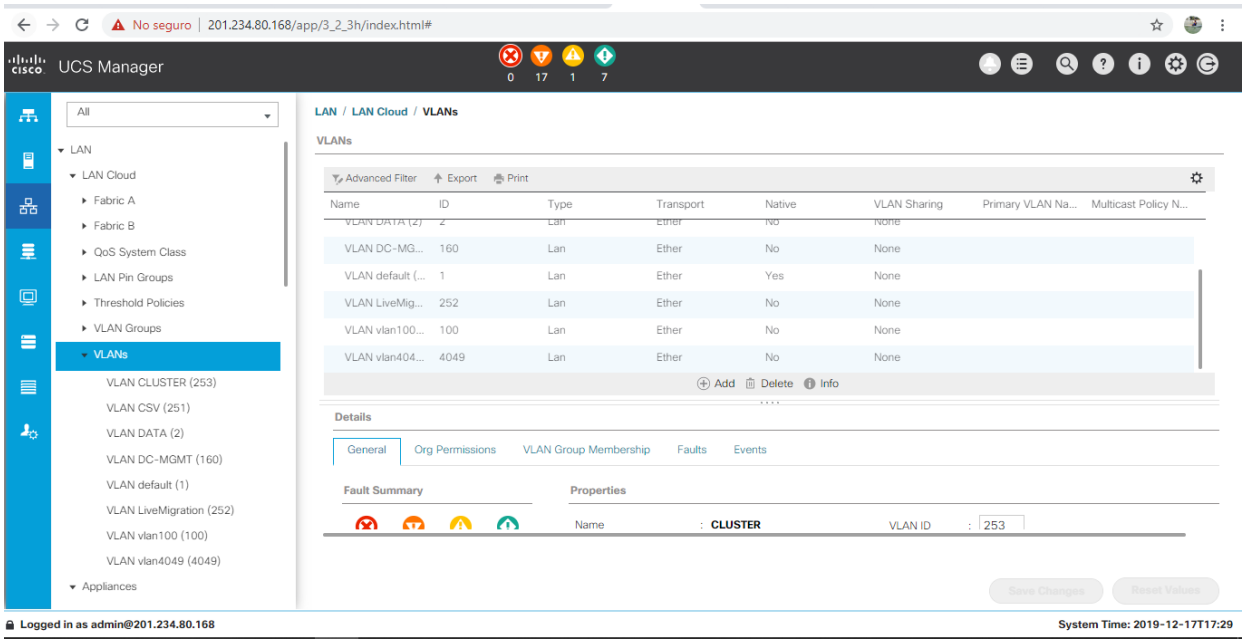


Figura 47. VLANs del UCS Manager.

En la Figura 48 se puede apreciar los detalles de la *VLAN 100* después de su creación.

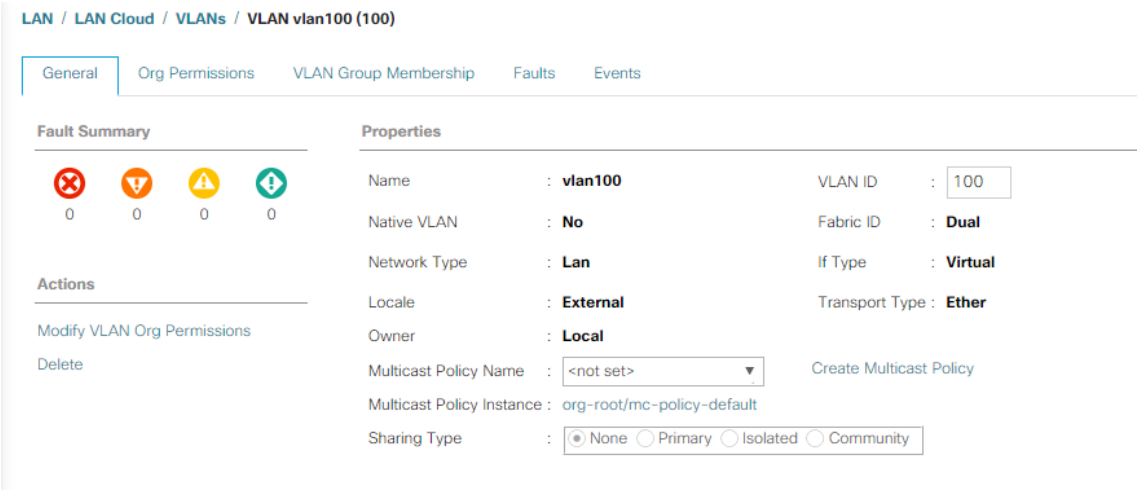


Figura 48. Datos de la creación de la *VLAN 100*.

En la Figura 49 se puede apreciar los detalles de la *VLAN 4049* después de su creación.

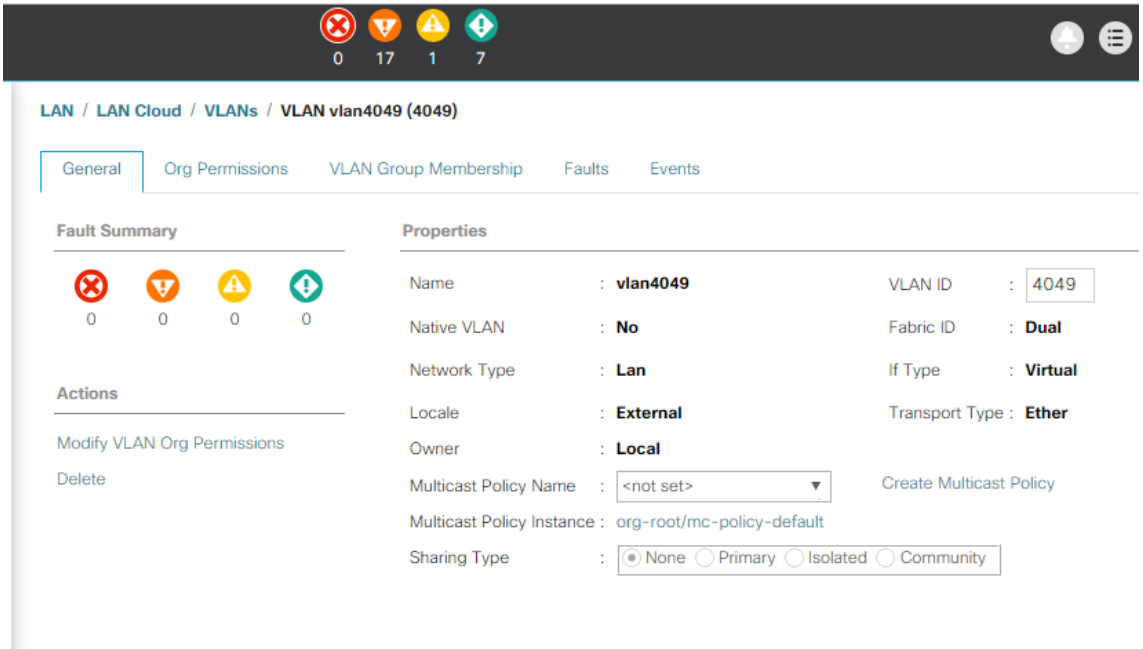


Figura 49. Datos de la creación de la *VLAN 4049*.

Otra prueba complementaria a la de crear *VLANs* que se realizó fue la de eliminación de *VLANs.* El procedimiento realizado es exactamente el mismo que en la creación de *VLANs,* la única diferencia es que cuando el usuario recibe el

menú de inicio por el asistente de voz Alexa, el usuario debe elegir la intención "*Remove V Lan*" seguida del número asignado a la *VLAN* que se desea eliminar.

Finalmente, la última prueba que se realizó fue la de creación de un perfil de servicio. Al igual que las anteriores pruebas el usuario debe iniciar la habilidad y el menú de inicio mediante la frase "*Alexa, ask devnet*". Después de que el asistente de voz termine de decir el menú el usuario debe iniciar la intención mediante la frase "*Provision a server*".

Como consecuencia se realiza un aprovisionamiento de un servidor dentro del UCS Manager. El asistente de voz Alexa, efectúa una respuesta al usuario con el número del servidor, el chassis y el nombre del perfil de servicio que se le asignó al servidor.

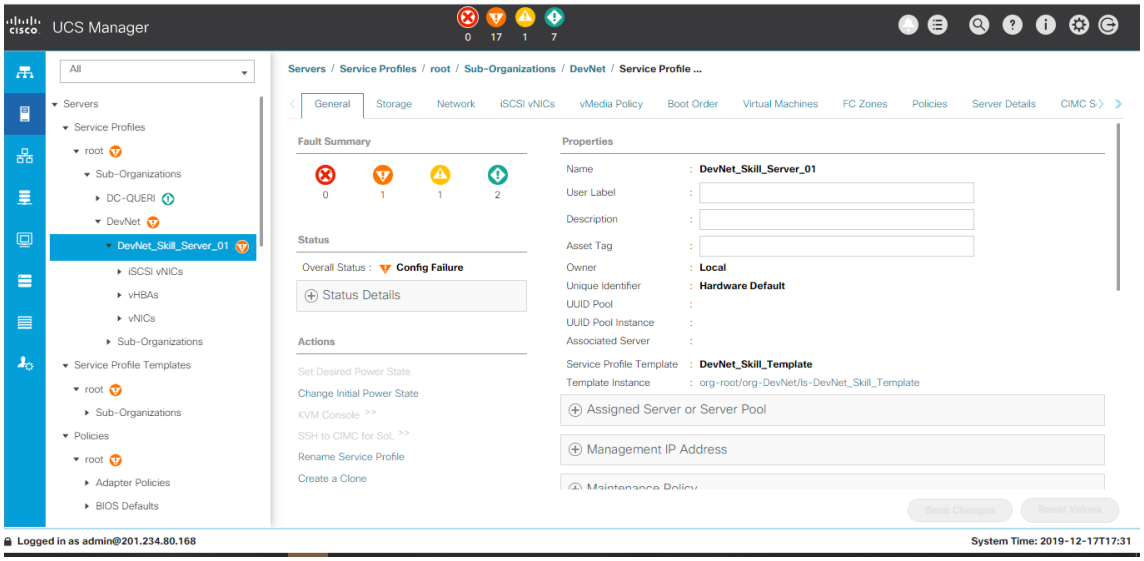


Figura 50. Creación de un Service Profile.

5. CONCLUSIONES Y RECOMENDACIONES

5.1 Conclusiones

El diagrama en el servicio *Lambda* lo fundamental es el permiso que se asigna a la función creada previamente. Ya que mediante eso se puede configurar los parámetros que va a tener la función. Ejemplo se da el permiso *S3* y funciones de Alexa.

La IP Pública se basa según la región en la que está configura la cuenta en *AWS* y se la puede buscar mediante un comando ejecutado en *PowerShell* que se encuentra en los anexos.

La versión de *Python* empleada en este proyecto es la 2.7. Cabe resaltar que esta versión quedará sin soporte en febrero del 2020 por lo que es recomendable emplear *Python* 3.7.

Esta implementación propuesta tiene como fin el poder motivar a los estudiantes a crear nuevos prototipos basados en esta nueva tecnología. Como se puede observar, además de brindar facilidad al usuario a la hora de trabajar, también se quiso orientar este proyecta al tema de la discapacidad.

Se debe usar el asistente de voz Alexa de Amazon, ya que al estar trabajando en las plataformas de Amazon Developer y Amazon Alexa con las mismas credenciales se realiza una vinculación automática a través de la nube.

DevNet ofrece una gran cantidad de laboratorios que permiten adaptar según la necesidad que tenga un administrador de red. Se enfoca en varios campos, y en dos en específico utiliza Amazon Alexa que es en el UCS Manager y es con la tecnología *Meraki* en *Access Point*.

5.2 Recomendaciones

Cuando se crea la cuenta en *AWS* se podrá usar después de 24 horas en caso de ser gratuita. Adicional solo se puede utilizar para los servicios que ofrece Amazon, por lo que se debe crear otra cuenta en Amazon Alex donde se encuentran las funcionalidades para crear las diferentes habilidades.

Al momento de sincronizar la habilidad creada en Amazon Alexa se debe especificar el idioma (inglés US) en el que se encuentra el código en *GitHub*. Ya que puede existir un conflicto de idiomas y Alexa no va a saber cuál elegir.

Especificar las credenciales de administrador del UCS Manager en el código que se réplica en *GitHub* ya que si no son las correctas va a dar error de conexión cuando se invoque un método por medio del comando de voz.

Crear una cuenta genérica en todas las plataformas involucradas para futuros proyectos que beneficien a los estudiantes interesados en alcanzar nuevos avances en este campo.

Los puertos utilizados para la salida y entrada de la IP Pública vinculada con el UCS Manager son HTTPS:443, HTTP:80, SSH:22.

Hay que tener en cuenta a la hora de ejecutar los diferentes scripts en *Power Shell*, se debe activar las políticas de ejecución mediante los comandos: `Get-ExecutionPolicy y Set-ExecutionPolicy Unrestricted`.

La incorporación y uso de los asistentes de voz por parte de los usuarios es mayor en la actualidad. Por lo que hacer ver a los usuarios que con los dispositivos de uso cotidiano se pueden realizar grandes proyectos y prototipos es de gran importancia para futuros profesionales.

REFERENCIAS

Alexa Web Information Service. (2019). Recuperado el 22 de octubre de 2019 de https://docs.aws.amazon.com/es_es/AlexaWebInfoService/latest/

Amazon Simple Storage Service: Guía de introducción. (2019). Recuperado el 22 de octubre de 2019 de https://docs.aws.amazon.com/es_es/AmazonS3/latest/gsg/s3-gsg.pdf

Amazon Echo Dot (3ª Generación) - Asistente personal Alexa. (2019). Recuperado el 22 de octubre de 2019 de https://www.droneymas.com/amazon-echo-dot-3a-generacion/

Automatización de Cisco UCS usando Python SDK. (2014). Recuperado el 22 de octubre de 2019 de https://sreeninet.wordpress.com/2014/09/20/cisco-ucs-automation-using-python-sdk/

AWS | Almacenamiento de datos seguro en la nube (S3). (2019). Recuperado el 22 de octubre de 2019 de https://aws.amazon.com/es/s3/

AWS Lambda: Guía para desarrolladores. (2019). Recuperado el 22 de octubre de 2019 de https://docs.aws.amazon.com/es_es/lambda/latest/dg/lambda-dg.pdf

AWS | Lambda - Gestión de recursos informáticos. (2019). Recuperado el 22 de octubre de 2019 de https://aws.amazon.com/es/lambda/

Castilleja, J., & Arriaga Alvarado, U. (2018). Recuperado el 20 de octubre de 2019 de https://www.ciscolive.com/c/dam/r/ciscolive/latam/docs/2018/pdf/DEVNET-2129-LG.pdf

Cisco DevNet: APIs, SDKs, Sandbox, and Community for Cisco Developers. (2019). Recuperado el 25 de octubre de 2019 de https://developer.cisco.com/

DevNet, red de desarrolladores de Cisco crece en Latino América. (2015). Recuperado el 23 de octubre de 2019 de https://americas.thecisconetwork.com/site/content/lang/es/id/4415

Gallardo Vázquez, S. (2019). Configuración de instalaciones domóticas y automáticas [Ebook] (2ª ed., Pp. 109-110). Madrid. Recuperado el 26 de octubre de 2019 de https://books.google.com.ec/books?id=iD2dDwAAQBAJ&pg=PA110&dq=amazon+echo+alexa&hl=es&sa=X&ved=0ahUKEwiK4pu4xbTjAhWTGc0KHQ9XD9cQ6AEIMDAC#v=onepage&q=amazon%20echo%20alexa&f=false

Garten, P. (2019). Guía de usuario de alexa [Ebook]. Babelcube Books. Recuperado el 01 de noviembre de 2019 de https://books.google.com.ec/books?id=aUClDwAAQBAJ&pg=PT40&dq=amazon+skills&hl=es&sa=X&ved=0ahUKEwih5Znpw4vlAhUsrlkKHVtDBWMQ6AEIJzAA#v=onepage&q=amazon%20skills&f=false

Guevara, C., & López, C. (2016). Recuperado el 30 de octubre de 2019 de https://bibdigital.epn.edu.ec/bitstream/15000/16832/1/CD-7411.pdf

Haack, W., Severance, M., Wallace, M., & Wohlwend, J. (2017). Security Analysis of the Amazon Echo. Recuperado el 22 de octubre de 2019 de https://pdfs.semanticscholar.org/35c8/47d63db1dd2c8cf36a3a8c3444cdeee605e4.pdf

Kuehl, K., & Zaatar, W. (2018). Recuperado el 20 de octubre de 2019 de https://www.netacad.com/sites/default/files/images/careers/Webinars/DevNet2/programmability_w_devnet_session_1.pdf

McDonough, J. Administración de cómputo UCS con Amazon Echo. Recuperado el 20 de octubre de 2019 de https://creations.devnetcloud.com/detail?cid=1e4a9f92-deea-11e7-9d2f-aed03b188287

Somasundaram, M. (2018). Setting up Alexa for Business with Cisco Telepresence video conferencing | Amazon Web Services. Recuperado el 20 de octubre de 2019 de https://aws.amazon.com/es/blogs/business-productivity/setting-up-alexa-for-business-with-cisco-telepresence-video-conferencing/

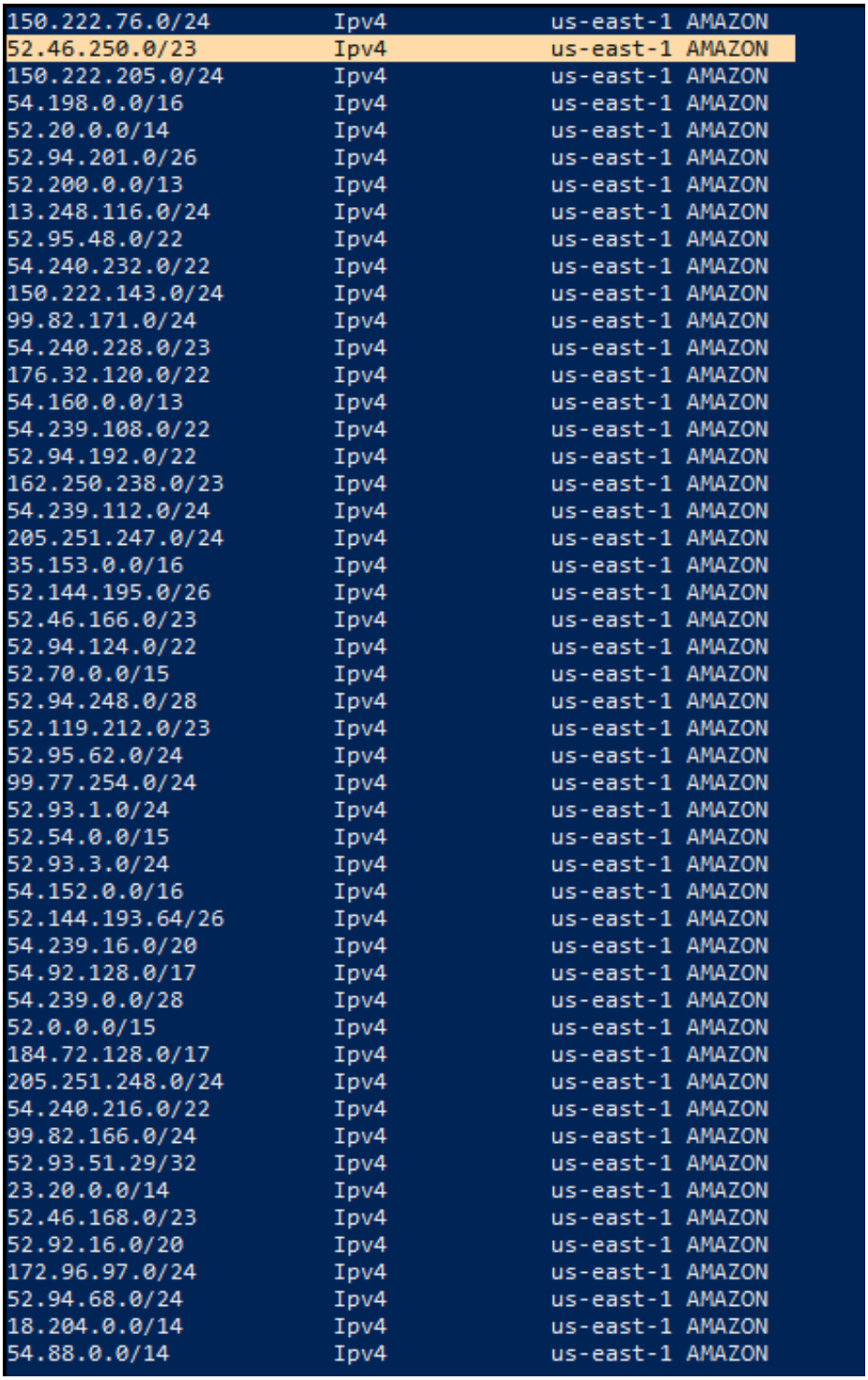
What Is Alexa for Business? - Alexa for Business. (2019). Recuperado el 22 de octubre de 2019 de https://docs.aws.amazon.com/es_es/a4b/latest/ag/what-is.html

ANEXOS

Rango de IPs públicas que ofrece Amazon para la región us-east-1.

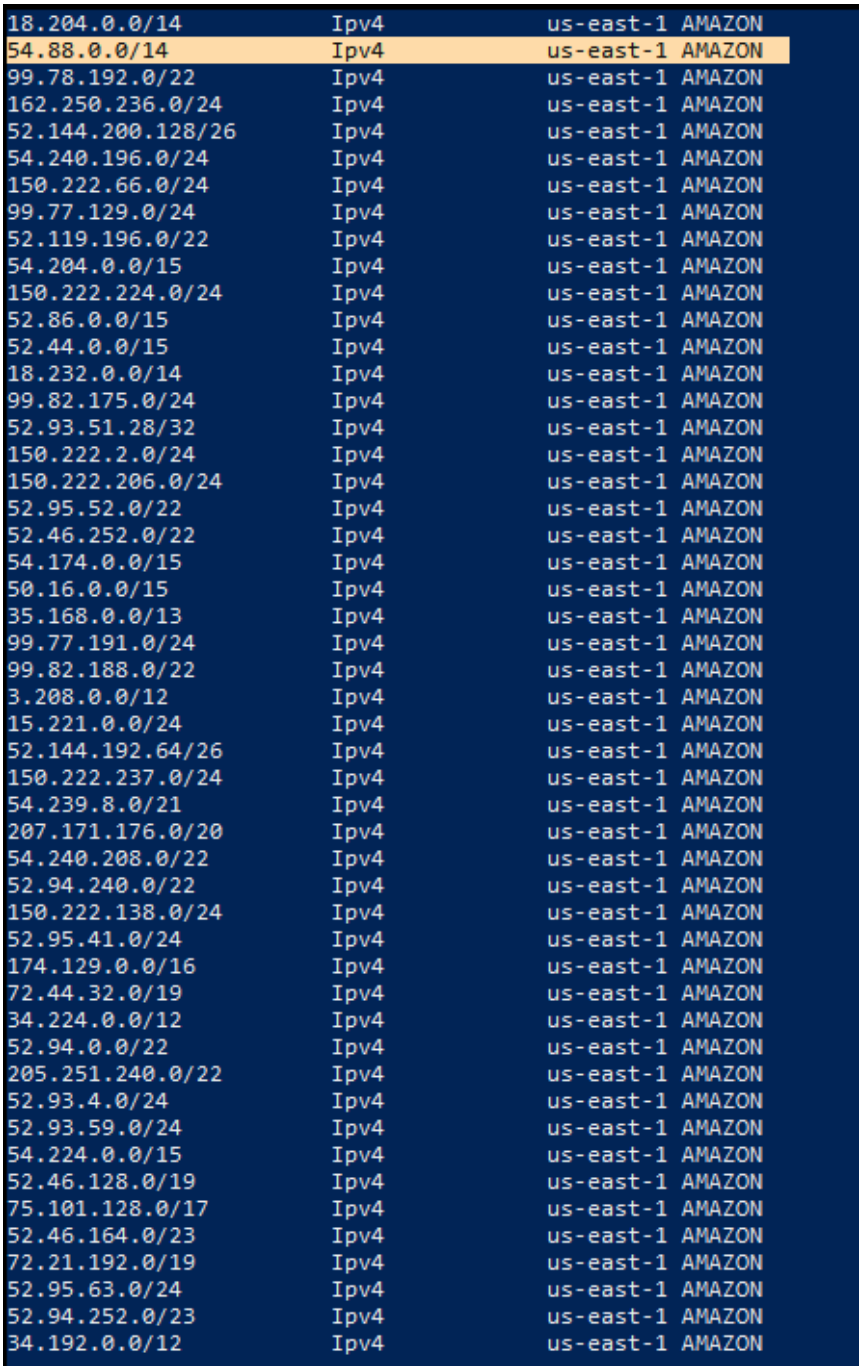
```
PS C:\WINDOWS\system32> Get-AWSPublicIpAddressRange -Region us-east-1 -ServiceKey AMAZON

IpPrefix              IpAddressFormat Region    Service
--------              --------------- ------    -------
18.208.0.0/13         Ipv4            us-east-1 AMAZON
52.95.245.0/24        Ipv4            us-east-1 AMAZON
54.196.0.0/15         Ipv4            us-east-1 AMAZON
216.182.224.0/21      Ipv4            us-east-1 AMAZON
52.119.224.0/21       Ipv4            us-east-1 AMAZON
216.182.232.0/22      Ipv4            us-east-1 AMAZON
13.248.103.0/24       Ipv4            us-east-1 AMAZON
52.144.193.128/26     Ipv4            us-east-1 AMAZON
107.20.0.0/14         Ipv4            us-east-1 AMAZON
52.94.224.0/20        Ipv4            us-east-1 AMAZON
99.77.128.0/24        Ipv4            us-east-1 AMAZON
150.222.71.0/24       Ipv4            us-east-1 AMAZON
67.202.0.0/18         Ipv4            us-east-1 AMAZON
205.251.246.0/24      Ipv4            us-east-1 AMAZON
199.127.232.0/22      Ipv4            us-east-1 AMAZON
52.93.249.0/24        Ipv4            us-east-1 AMAZON
207.171.160.0/20      Ipv4            us-east-1 AMAZON
184.73.0.0/16         Ipv4            us-east-1 AMAZON
52.93.60.0/24         Ipv4            us-east-1 AMAZON
150.222.136.0/24      Ipv4            us-east-1 AMAZON
3.80.0.0/12           Ipv4            us-east-1 AMAZON
54.80.0.0/13          Ipv4            us-east-1 AMAZON
3.224.0.0/12          Ipv4            us-east-1 AMAZON
52.144.192.192/26     Ipv4            us-east-1 AMAZON
54.221.0.0/16         Ipv4            us-east-1 AMAZON
54.240.202.0/24       Ipv4            us-east-1 AMAZON
54.156.0.0/14         Ipv4            us-east-1 AMAZON
54.236.0.0/15         Ipv4            us-east-1 AMAZON
52.93.76.0/24         Ipv4            us-east-1 AMAZON
52.144.194.0/26       Ipv4            us-east-1 AMAZON
54.226.0.0/15         Ipv4            us-east-1 AMAZON
162.250.237.0/24      Ipv4            us-east-1 AMAZON
52.90.0.0/15          Ipv4            us-east-1 AMAZON
100.24.0.0/13         Ipv4            us-east-1 AMAZON
52.95.216.0/22        Ipv4            us-east-1 AMAZON
52.119.232.0/21       Ipv4            us-east-1 AMAZON
54.231.244.0/22       Ipv4            us-east-1 AMAZON
150.222.99.0/24       Ipv4            us-east-1 AMAZON
205.251.244.0/23      Ipv4            us-east-1 AMAZON
54.231.0.0/17         Ipv4            us-east-1 AMAZON
52.144.192.0/26       Ipv4            us-east-1 AMAZON
54.210.0.0/15         Ipv4            us-east-1 AMAZON
150.222.76.0/24       Ipv4            us-east-1 AMAZON
52.46.250.0/23        Ipv4            us-east-1 AMAZON
```

Figura 51. Rango de IPs ipv4 de Amazon.



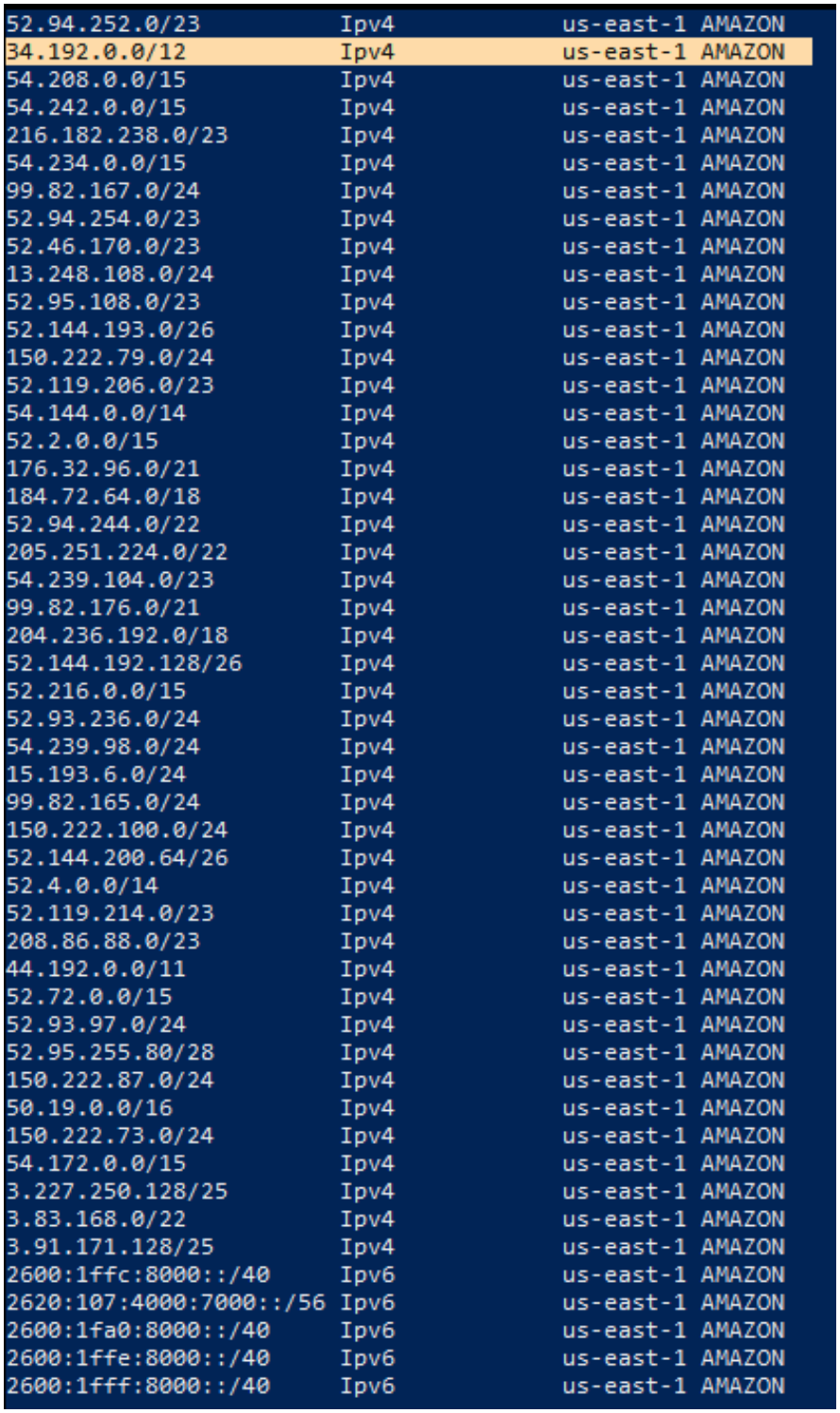
150.222.76.0/24	Ipv4	us-east-1 AMAZON
52.46.250.0/23	Ipv4	us-east-1 AMAZON
150.222.205.0/24	Ipv4	us-east-1 AMAZON
54.198.0.0/16	Ipv4	us-east-1 AMAZON
52.20.0.0/14	Ipv4	us-east-1 AMAZON
52.94.201.0/26	Ipv4	us-east-1 AMAZON
52.200.0.0/13	Ipv4	us-east-1 AMAZON
13.248.116.0/24	Ipv4	us-east-1 AMAZON
52.95.48.0/22	Ipv4	us-east-1 AMAZON
54.240.232.0/22	Ipv4	us-east-1 AMAZON
150.222.143.0/24	Ipv4	us-east-1 AMAZON
99.82.171.0/24	Ipv4	us-east-1 AMAZON
54.240.228.0/23	Ipv4	us-east-1 AMAZON
176.32.120.0/22	Ipv4	us-east-1 AMAZON
54.160.0.0/13	Ipv4	us-east-1 AMAZON
54.239.108.0/22	Ipv4	us-east-1 AMAZON
52.94.192.0/22	Ipv4	us-east-1 AMAZON
162.250.238.0/23	Ipv4	us-east-1 AMAZON
54.239.112.0/24	Ipv4	us-east-1 AMAZON
205.251.247.0/24	Ipv4	us-east-1 AMAZON
35.153.0.0/16	Ipv4	us-east-1 AMAZON
52.144.195.0/26	Ipv4	us-east-1 AMAZON
52.46.166.0/23	Ipv4	us-east-1 AMAZON
52.94.124.0/22	Ipv4	us-east-1 AMAZON
52.70.0.0/15	Ipv4	us-east-1 AMAZON
52.94.248.0/28	Ipv4	us-east-1 AMAZON
52.119.212.0/23	Ipv4	us-east-1 AMAZON
52.95.62.0/24	Ipv4	us-east-1 AMAZON
99.77.254.0/24	Ipv4	us-east-1 AMAZON
52.93.1.0/24	Ipv4	us-east-1 AMAZON
52.54.0.0/15	Ipv4	us-east-1 AMAZON
52.93.3.0/24	Ipv4	us-east-1 AMAZON
54.152.0.0/16	Ipv4	us-east-1 AMAZON
52.144.193.64/26	Ipv4	us-east-1 AMAZON
54.239.16.0/20	Ipv4	us-east-1 AMAZON
54.92.128.0/17	Ipv4	us-east-1 AMAZON
54.239.0.0/28	Ipv4	us-east-1 AMAZON
52.0.0.0/15	Ipv4	us-east-1 AMAZON
184.72.128.0/17	Ipv4	us-east-1 AMAZON
205.251.248.0/24	Ipv4	us-east-1 AMAZON
54.240.216.0/22	Ipv4	us-east-1 AMAZON
99.82.166.0/24	Ipv4	us-east-1 AMAZON
52.93.51.29/32	Ipv4	us-east-1 AMAZON
23.20.0.0/14	Ipv4	us-east-1 AMAZON
52.46.168.0/23	Ipv4	us-east-1 AMAZON
52.92.16.0/20	Ipv4	us-east-1 AMAZON
172.96.97.0/24	Ipv4	us-east-1 AMAZON
52.94.68.0/24	Ipv4	us-east-1 AMAZON
18.204.0.0/14	Ipv4	us-east-1 AMAZON
54.88.0.0/14	Ipv4	us-east-1 AMAZON

Figura 52. Rango de IPs ipv4 de Amazon.



18.204.0.0/14	Ipv4	us-east-1 AMAZON
54.88.0.0/14	Ipv4	us-east-1 AMAZON
99.78.192.0/22	Ipv4	us-east-1 AMAZON
162.250.236.0/24	Ipv4	us-east-1 AMAZON
52.144.200.128/26	Ipv4	us-east-1 AMAZON
54.240.196.0/24	Ipv4	us-east-1 AMAZON
150.222.66.0/24	Ipv4	us-east-1 AMAZON
99.77.129.0/24	Ipv4	us-east-1 AMAZON
52.119.196.0/22	Ipv4	us-east-1 AMAZON
54.204.0.0/15	Ipv4	us-east-1 AMAZON
150.222.224.0/24	Ipv4	us-east-1 AMAZON
52.86.0.0/15	Ipv4	us-east-1 AMAZON
52.44.0.0/15	Ipv4	us-east-1 AMAZON
18.232.0.0/14	Ipv4	us-east-1 AMAZON
99.82.175.0/24	Ipv4	us-east-1 AMAZON
52.93.51.28/32	Ipv4	us-east-1 AMAZON
150.222.2.0/24	Ipv4	us-east-1 AMAZON
150.222.206.0/24	Ipv4	us-east-1 AMAZON
52.95.52.0/22	Ipv4	us-east-1 AMAZON
52.46.252.0/22	Ipv4	us-east-1 AMAZON
54.174.0.0/15	Ipv4	us-east-1 AMAZON
50.16.0.0/15	Ipv4	us-east-1 AMAZON
35.168.0.0/13	Ipv4	us-east-1 AMAZON
99.77.191.0/24	Ipv4	us-east-1 AMAZON
99.82.188.0/22	Ipv4	us-east-1 AMAZON
3.208.0.0/12	Ipv4	us-east-1 AMAZON
15.221.0.0/24	Ipv4	us-east-1 AMAZON
52.144.192.64/26	Ipv4	us-east-1 AMAZON
150.222.237.0/24	Ipv4	us-east-1 AMAZON
54.239.8.0/21	Ipv4	us-east-1 AMAZON
207.171.176.0/20	Ipv4	us-east-1 AMAZON
54.240.208.0/22	Ipv4	us-east-1 AMAZON
52.94.240.0/22	Ipv4	us-east-1 AMAZON
150.222.138.0/24	Ipv4	us-east-1 AMAZON
52.95.41.0/24	Ipv4	us-east-1 AMAZON
174.129.0.0/16	Ipv4	us-east-1 AMAZON
72.44.32.0/19	Ipv4	us-east-1 AMAZON
34.224.0.0/12	Ipv4	us-east-1 AMAZON
52.94.0.0/22	Ipv4	us-east-1 AMAZON
205.251.240.0/22	Ipv4	us-east-1 AMAZON
52.93.4.0/24	Ipv4	us-east-1 AMAZON
52.93.59.0/24	Ipv4	us-east-1 AMAZON
54.224.0.0/15	Ipv4	us-east-1 AMAZON
52.46.128.0/19	Ipv4	us-east-1 AMAZON
75.101.128.0/17	Ipv4	us-east-1 AMAZON
52.46.164.0/23	Ipv4	us-east-1 AMAZON
72.21.192.0/19	Ipv4	us-east-1 AMAZON
52.95.63.0/24	Ipv4	us-east-1 AMAZON
52.94.252.0/23	Ipv4	us-east-1 AMAZON
34.192.0.0/12	Ipv4	us-east-1 AMAZON

Figura 53. Rango de IPs ipv4 de Amazon.



52.94.252.0/23	Ipv4	us-east-1 AMAZON
34.192.0.0/12	Ipv4	us-east-1 AMAZON
54.208.0.0/15	Ipv4	us-east-1 AMAZON
54.242.0.0/15	Ipv4	us-east-1 AMAZON
216.182.238.0/23	Ipv4	us-east-1 AMAZON
54.234.0.0/15	Ipv4	us-east-1 AMAZON
99.82.167.0/24	Ipv4	us-east-1 AMAZON
52.94.254.0/23	Ipv4	us-east-1 AMAZON
52.46.170.0/23	Ipv4	us-east-1 AMAZON
13.248.108.0/24	Ipv4	us-east-1 AMAZON
52.95.108.0/23	Ipv4	us-east-1 AMAZON
52.144.193.0/26	Ipv4	us-east-1 AMAZON
150.222.79.0/24	Ipv4	us-east-1 AMAZON
52.119.206.0/23	Ipv4	us-east-1 AMAZON
54.144.0.0/14	Ipv4	us-east-1 AMAZON
52.2.0.0/15	Ipv4	us-east-1 AMAZON
176.32.96.0/21	Ipv4	us-east-1 AMAZON
184.72.64.0/18	Ipv4	us-east-1 AMAZON
52.94.244.0/22	Ipv4	us-east-1 AMAZON
205.251.224.0/22	Ipv4	us-east-1 AMAZON
54.239.104.0/23	Ipv4	us-east-1 AMAZON
99.82.176.0/21	Ipv4	us-east-1 AMAZON
204.236.192.0/18	Ipv4	us-east-1 AMAZON
52.144.192.128/26	Ipv4	us-east-1 AMAZON
52.216.0.0/15	Ipv4	us-east-1 AMAZON
52.93.236.0/24	Ipv4	us-east-1 AMAZON
54.239.98.0/24	Ipv4	us-east-1 AMAZON
15.193.6.0/24	Ipv4	us-east-1 AMAZON
99.82.165.0/24	Ipv4	us-east-1 AMAZON
150.222.100.0/24	Ipv4	us-east-1 AMAZON
52.144.200.64/26	Ipv4	us-east-1 AMAZON
52.4.0.0/14	Ipv4	us-east-1 AMAZON
52.119.214.0/23	Ipv4	us-east-1 AMAZON
208.86.88.0/23	Ipv4	us-east-1 AMAZON
44.192.0.0/11	Ipv4	us-east-1 AMAZON
52.72.0.0/15	Ipv4	us-east-1 AMAZON
52.93.97.0/24	Ipv4	us-east-1 AMAZON
52.95.255.80/28	Ipv4	us-east-1 AMAZON
150.222.87.0/24	Ipv4	us-east-1 AMAZON
50.19.0.0/16	Ipv4	us-east-1 AMAZON
150.222.73.0/24	Ipv4	us-east-1 AMAZON
54.172.0.0/15	Ipv4	us-east-1 AMAZON
3.227.250.128/25	Ipv4	us-east-1 AMAZON
3.83.168.0/22	Ipv4	us-east-1 AMAZON
3.91.171.128/25	Ipv4	us-east-1 AMAZON
2600:1ffc:8000::/40	Ipv6	us-east-1 AMAZON
2620:107:4000:7000::/56	Ipv6	us-east-1 AMAZON
2600:1fa0:8000::/40	Ipv6	us-east-1 AMAZON
2600:1ffe:8000::/40	Ipv6	us-east-1 AMAZON
2600:1fff:8000::/40	Ipv6	us-east-1 AMAZON

Figura 54. Rango de IPs ipv4 e ipv6 de Amazon.

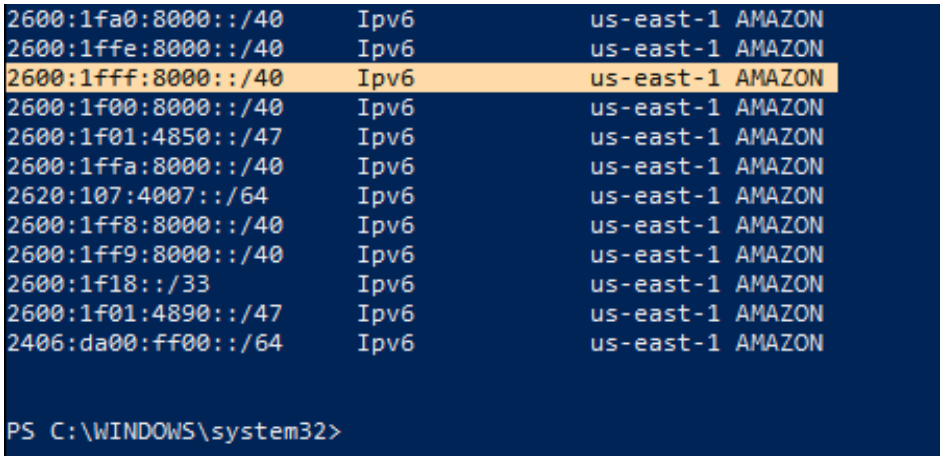


Figura 55. Rango de IPs ipv6 de Amazon.