ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELECTRÍCA Y ELECTRÓNICA

AUTOMATIZACIÓN DE REDES PARA IOT

ESTUDIO INTRODUCTORIO DE LA PLATAFORMA DE ORQUESTACIÓN DE REDES NSO

TRABAJO DE INTEGRACIÓN CURRICULAR PRESENTADO COMO REQUISITO PARA LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN TELECOMUNICACIONES

LUIS EDUARDO BASTIDAS REINOSO

luis.bastidas@epn.edu.ec

DIRECTOR: M.Sc CARLOS ROBERTO EGAS ACOSTA

carlos.egas@epn.edu.ec

DMQ, agosto 2023

CERTIFICACIONES

Yo, LUIS EDUARDO BASTIDAS REINOSO declaro que el trabajo de integración curricular aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

_____

LUIS EDUARDO BASTIDAS REINOSO

Certifico que el presente trabajo de integración curricular fue desarrollado por CARLOS ROBERTO EGAS ACOSTA, bajo mi supervisión.

_____

M.Sc CARLOS ROBERTO EGAS ACOSTA

DIRECTOR

DECLARACIÓN DE AUTORÍA

A través de la presente declaración, afirmamos que el trabajo de integración curricular aquí descrito, así como el producto resultante del mismo, son públicos y estarán a disposición de la comunidad a través del repositorio institucional de la Escuela Politécnica Nacional; sin embargo, la titularidad de los derechos patrimoniales nos corresponde a los autores que hemos contribuido en el desarrollo del presente trabajo; observando para el efecto las disposiciones establecidas por el órgano competente en propiedad intelectual, la normativa interna y demás normas.

LUIS EDUARDO BASTIDAS REINOSO

M.Sc CARLOS ROBERTO EGAS ACOSTA

DEDICATORIA

Dedicado a mí mismo, y a todos que estuvieron en el momento adecuado…

Hoy, recordando hace algún tiempo atrás todos los pasos dados, los obstáculos superados y todo el esfuerzo constante y la determinación para llegar a este punto de mi vida académica.

A lo largo de este proceso, me he enfrentado a muchos desafíos que me pusieron a prueba y en ciertas ocasiones a tirar la toalla, pero en cada momento de duda, encontré la fuerza para continuar y la convicción para no rendirme, las cuales me ayudaron a crecer y aprender y con ellos definir la clase de profesional que soy actualmente.

Gracias a mi ser perseverante, el nunca perder de vista el objetivo y mantenerme enfocado en el camino, a mi esfuerzo incansable, que me llevó a estudiar horas interminables, leer innumerables artículos y trabajar arduamente para lograr cada avance en mi vida. Cada pequeño paso fue una contribución significativa hacia la culminación de este logro.

Me siento agradecido y dedico está tesis, a todas las personas que estuvieron en todo mi proceso académico y que supieron estar ahí en su momento, dándome su apoyo, consejos y ánimos para poder llegar a este meta.

Esta tesis es el testimonio de mi dedicación y amor por el aprendizaje, y me recuerda que soy capaz de superar cualquier desafío que la vida presente en el futuro.

Con cariño y gratitud,

Luis Eduardo Bastidas Reinoso.

AGRADECIMIENTO

Quiero expresar mi más sincero agradecimiento a mis padres, por su amor incondicional, apoyo constante y sacrificio a lo largo de este viaje académico. Su aliento y dedicación han sido mi fuente de inspiración para alcanzar este logro. Gracias por creer en mí y por ser mi principal motivación.

También deseo extender mi gratitud a mis maestros y profesores, quienes me han guiado con su conocimiento experto y han compartido valiosas enseñanzas a lo largo de mi investigación. Sus consejos, críticas constructivas y orientación han sido fundamentales para dar forma a este trabajo y para mi desarrollo personal y académico.

Cada uno de ustedes ha dejado una huella imborrable en mi trayectoria, y les estoy sinceramente agradecido por su impacto en mi formación.

Con cariño y gratitud,

Luis Eduardo Bastidas Reinoso.

ÍNDICE DE CONTENIDO

ÍNDICE DE IMÁGENES

ÍNDICE DE TABLAS

RESUMEN

En el presente trabajo de Integración Curricular se realiza el proceso de análisis y verificación de funcionamiento del laboratorio de Cisco sobre el manejo de NSO (Servicio de Orquestación de Redes). La base del desarrollo de este documento es establecer la configuración de una red con varios dispositivos CISCO y puntualizar como se relacionan las diferentes herramientas con NSO, que son estudiadas en el capítulo 2.2.

El capítulo 2.1 tiene como objetivo presentar los comando básicos y más usados dentro de la configuración de NSO (Servicio de Orquestación de Redes) y la utilización de las librerías y archivos que se utilizaran para la configuración, al ser un laboratorio habrá librerías predefinidas ya incorporadas en el directorio NSC.

En el capítulo 2.2 se detalla ya las configuraciones de NSO (Servicio de Orquestación de Redes) para una red diseña por el laboratorio, la información de la red es proporcionada por el laboratorio de CISCO con las respectivas características de los dispositivos a que grupo se van a relacionar dentro de NSO, las respectivas direcciones IP's, tipo de comunicación, interfaces y puertos a utilizarse.

En el capítulo 2.3 se considera la implementación de Python de CISCO para poder realizar configuraciones en NSO, integrando la plataforma con otros sistemas y herramientas de administración y monitorización de red, facilitando la interoperabilidad y la creación de soluciones personalizada, es decir realizar operaciones CRUD (Crear, Leer, Actualizar y Eliminar) en los datos YANG.

PALABRAS CLAVE: NSO, YANG,PYTHON, CISCO

ABSTRACT

In the present work of Curricular Integration, the process of analysis and verification of the operation of the Cisco laboratory on the management of NSO (Network Orchestration Service) is conducted. The basis of the development of this document is to establish the configuration of a network with several CISCO devices and point out how the different tools are related to NSO, which are studied in chapter 2.2.

Chapter 2.1 aims to present the basic and most used commands within the NSO (Network Orchestration Service) configuration and the use of the libraries and files that will be used for the configuration, being a laboratory there will be predefined libraries already incorporated in the NSC directory.

Chapter 2.2 details the NSO (Network Orchestration Service) configurations for a network designed by the laboratory, the network information is provided by the CISCO laboratory with the respective characteristics of the devices to which group they will go. to relate within NSO, the respective IP addresses, type of communication, interfaces, and ports to be used.

In chapter 2.3 the implementation of CISCO's Python is considered to be able to carry out configurations in NSO, integrating the platform with other systems and tools for network administration and monitoring, facilitating interoperability and the creation of customized solutions, that is, performing CRUD operations (Create, Read, Update and Delete) on the YANG data.

KEYWORDS: NSO,YANG,PYTHON;CISCO

1. INTRODUCCIÓN

Actualmente la automatización de los servidores y servicios, generan un gran conflicto a la hora de dar una solución rápida y con un gasto mínimo a los diferentes clientes dentro de las redes, la entrega de servicios siempre es una molestia: agregar, cambiar y eliminar servicios exige una codificación personalizada o interrupciones del servicio. NSO es una aplicación de Linux que organiza el ciclo de vida de configuración de los dispositivos de red físicos y virtuales, permite dar soluciones rápidas automatizando la gestión integral del ciclo de vida de los servicios en entornos físicos y virtualizados. Aprovechando el poder de YANG y NETCONF, la solución Cisco NSO simplifica y fortalecer la gestión de servicios. Se puede usar para abordar la virtualización de funciones de red (NFV), administración y orquestación (MANO), redes definidas por software (SDN), así como su red física tradicional y todos sus componentes.

En el presente trabajo se realizará un análisis y configuración del laboratorio virtual que propone CISCO para la orquestación de servicios de red (NSO), tomar las ventajas y desventajas que otorga el manejo de un sistema que permite la automatización y respuesta rápida a los escenarios que se puede presentar en los diferentes servicios que puede poseer uno o varios clientes, así como también la integración de sus redes.

Permitiendo de igual manera para los administradores de red a localizar de manera rápida cualquier error o inconveniente presentado en la red que se maneja dentro de la NSO, y de igual manera realizar un proceso más rápido de configuración al incorporar servidores con características similares a los servidores que ya se encuentra dentro de la red y que son contralado mediante la NSO; agilitando los procesos de respuesta y manejo dentro de la una red, permitiendo que sea imperceptibles los cambios o configuraciones para el usuario final.

Gracias al enfoque que tiene NSO basado en modelos y su capacidad multi-vendor lo hacen valioso tanto para proveedores de servicios como para empresas que buscan simplificar y mejorar la gestión de su infraestructura de red, mediante la automatización reduce el tiempo dedicado a tareas manuales y repetitivas, lo que libera a los equipos para centrarse en actividades más estratégicas, reducción costos de manejo por los administradores estos se logra al reducir los errores y mejorar la eficiencia.

1.1 Objetivo general

Analizar, configurar y verificar el control de acceso del simulador de Cisco NSO.

1.2 Objetivos específicos

El presente trabajo tiene los siguientes objetivos específicos:

1. Llevar a cabo un estudio de los diferentes programas y herramientas que usa NSO.
2. Verificar las funcionalidades del simulador de NSO con respecto a los servicios.
3. Analizar el uso de los diferentes programas (Yang, Python, entre otros), en la orquestación de servicios de red NSO

1.3 Alcance

La contribución del presente trabajo se centra en el análisis, configuración y manejo de la orquestación de servicios de red NSO CISCO. Para esto se utilizará el simulador propio de Cisco, que presenta una topología con diferentes servidores y servicios.

Para el manejo y configuración del simulador se debe tener conocimiento previo y básico sobre:

- Línea de comandos de sistemas operativos tipo UNIX
- Protocolo de configuración de red (NETCONF)
- Modelado de datos Yet Another Next Generation (YANG)
- Desarrollo de software Python.

Además, completado este laboratorio, se podrá:

Configure una instancia de NSO desde una instalación local.

Conéctese a dispositivos de red con NSO, configurándolos y envió de comandos

Usar NSO con algunos casos de uso básicos.

Se realizarán pruebas y mediciones básicas dentro del simulador; evaluar los posibles escenarios que presente. Al finalizar con las configuraciones y análisis del simulador; se verificarán las ventajas y desventajas.[1]

1.4 Marco Teórico

1.1.1 VPN

VPN (Virtual Private Network – Red Privada Virtual) es una tecnología que permite la creación de conexiones seguras y cifradas entre dispositivos o redes a través de internet mediante una red pública. La conexión cifrada tiene como objetivo proteger la privacidad y la seguridad de los datos transmitidos entre los puntos conectados, evita que personas no autorizadas espíen el tráfico y permite al usuario realizar el trabajo de forma remota, en la Figura1 se observar cómo se realiza el proceso de conexión por VPN. [2]

Ya que la comunicación entre el dispositivo y la red está encriptada, cualquier información transmitida permanece confidencial durante su trayecto. Un empleado puede trabajar fuera de la oficina y aun así conectarse de forma segura a la red corporativa. Incluso los teléfonos inteligentes y las tabletas pueden conectarse a través de una VPN. [3]

Las principales funciones de una VPN son:

1. Privacidad: Oculta la dirección IP real y enmascara la ubicación, lo que dificulta que terceros rastreen las actividades entre los puntos conectados.
2. Seguridad: Los datos transmitidos están encriptados (cifrados), lo que protege la información de posibles ciberataques o hackers.

Tipos de VPN

1. Acceso remoto. – permite conectar de forma segura un dispositivo (computadoras portátiles, tabletas o teléfonos inteligentes), que estén fuera de una red privada, estos dispositivos se conocen como terminales, visto desde la red interna son como un dispositivo más en la red. Los avances tecnológicos han permitido realizar comprobaciones en los puntos finales para asegurarse de que se cumplan ciertas normativas antes de conectarse por la VPN. Como al Figura 1.1
2. Sitio a sitio. – permite conectar la oficina corporativa con las sucursales a través de Internet. Las VPN de sitio a sitio se utilizan cuando la distancia hace que sea poco práctico tener conexiones de red directas entre estas oficinas.[4]

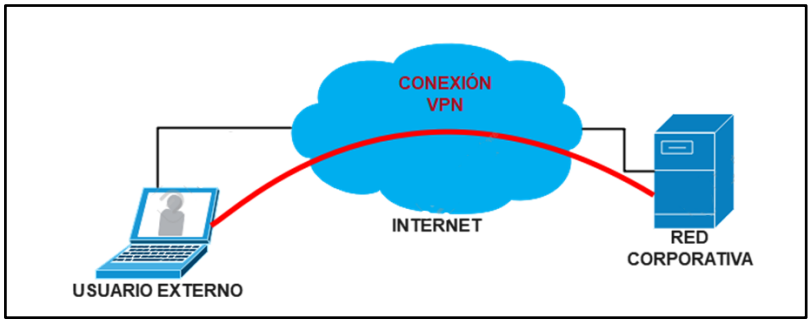

Figura 1.1 Diseño de una red con VPN

1.1.2 MOBAXTERM

Es una herramienta de software robusto y versátil que brinda una solución y administración remota de terminal, además, de una suite de funciones robusta para sistemas Windows. Es muy popular entre los profesionales de TI y desarrolladores, debido a sus numerosas funciones y fácil manejo, ya que combina un terminal de línea de comando integrado con características adicionales que permiten a los usuarios acceder y administrar de manera eficiente los sistemas remotos. [5]

Permite el manejo de una amplia variedad de protocolos de red, como SSH, Telnet, RDP, VNC, FTP, SFTP, entre otros, además ofrece una solución completa y segura para conectarse remotamente con servidores interno y externos, permite la ejecución de comandos de forma confiable, todo desde una interfaz centralizada y amigable al usuario.[5]

Las características de MobaXterm incluyen:

1. X11 Server: Incluye el servidor X11 para permitir la ejecución de aplicaciones gráficas de Linux y Unix en un entorno Windows.
2. Herramientas de red: Permite una variedad de herramientas de red como ping, escáner de puertos, traceroute, entre otros.
3. Sesiones múltiples: Permite administrar múltiples sesiones de terminal en pestañas para una experiencia de usuario más organizada.
4. Gestor de sesiones: Facilita la gestión de sesiones y conexiones guardadas para un acceso rápido y sencillo.
5. Personalización: Los usuarios tienen la capacidad de ajustar la apariencia y la configuración de acuerdo con sus propias preferencias.

6. Extensibilidad: Es posible agregar complementos y extensiones para ampliar sus capacidades.

1.1.3 Sistemas operativos Unix y linux (líneas de comando por consola).

Linux es un sistema operativo disponible de forma gratuita en Internet, que siempre en constate desarrollo y actualización por una comunidad de código abierto que tiene una colaboración a nivel internacional. Existen algunos proyectos que son de código cerrado, esto dedicada para el sector netamente comercial, como es el caso de Unix.[6]

Se han generado una cantidad versiones de Linux con expansiones del kernel que permite tener un sistema operativo completo, a todas estas versiones se las conoce como distribuciones (Alma-Linux, Ubuntu, Debian, CentOS, entre otros producido. A menudo, las distribuciones de Linux se desarrollan con un enfoque de aplicación concretas.

Linux suele equipararse a Unix o se conoce como un sistema operativo basado en Unix, entre otras cosas porque fue diseñado como un sistema similar a Unix, contiene funciones, comandos típicos de Unix. [7]

Los comandos de Linux son instrucciones que se pueden ingresar en la línea de comandos o en el terminal para interactuar con el sistema y realizar diversas tareas. Estos comandos permiten a los usuarios ejecutar acciones específicas, administrar archivos, directorios, procesos y configuraciones del sistema, entre otras cosas. Los comandos de Linux se componen de una palabra o letras seguidas de opciones y argumentos que modifican su comportamiento. Los comandos son sensibles a mayúsculas y minúsculas, lo que significa que "comando" y "COMANDO" son tratados como dos comandos distintos. [6]

Aquí hay algunos ejemplos típicos de categorías de comandos de Linux y sus usos:

1. **Gestión de archivos y directorios:**[8]
    - **ls:** Lista los archivos y carpetas en el directorio actual.
    - **cd**: Cambia el directorio actual.
    - **mkdir**: Crea un nuevo directorio.
    - **cp**: Copia archivos
    - **mv**: Mueve o renombra archivos y directorios.
2. **Visualización y manipulación de contenido:**[8]
    - **cat**: Muestra el contenido de un archivo.

- **vim:** Permite visualizar y editar un archivo.
- **tail**: Muestra las últimas líneas de un archivo.
- **grep**: Busca patrones en archivos o texto.

3. **Gestión de procesos:** [8]
    - **ps**: Muestra los procesos en ejecución.
    - **kill**: Termina un proceso en ejecución.
4. **Gestión de usuarios y permisos:** [8]
    - **sudo**: Ejecuta un comando con privilegio
    - **useradd**: Agrega un nuevo usuario al sistema.
    - **passwd**: Cambia la contraseña de un usuario.
    - **chmod**: Cambia los
    - **chown**: Cambia el propietario de un archivo o directorio.
5. **Información del sistema:**[8]
    - **uname**: Muestra información del sistema, como el nombre del kernel.
    - **df**: Muestra el espacio disponible en el sistema de archivos.
    - **free**: Muestra las características del dispositivo (RAM; procesador. entre otros)
    - **top** o **htop**: Muestra información en tiempo real sobre los procesos en ejecución y el uso de recursos del sistema.

1.1.4 Orquestador de servicio de redes (NSO)

En esencia, la orquestación de servicios de red busca simplificar y agilizar la administración de la red mediante la automatización de tareas repetitivas y la coordinación de múltiples componentes y servicios como se muestra en la Figura 1.2. En entornos de red tradicionales, la configuración y gestión de servicios de red, como enrutamiento, seguridad, balanceo de carga, firewall, etc., se realizó manualmente por parte de administradores de red. [9] [10]

Esto puede ser propenso a errores, lento y difícil de escalar en entornos grandes y dinámicos, con la orquestación de servicios de red, se utilizan herramientas y plataformas

de seguridad que permiten a los administradores definir políticas, reglas y configuraciones de red en forma de código o scripts. Estas herramientas pueden coordinar y aplicar estos scripts en varios dispositivos y componentes de red, lo que facilita la configuración y el mantenimiento coherente de la red.[10]

Figura 1.2 Diagrama de Orquestación de Redes*[10]*

Las tecnologías de virtualización y la nube han impulsado la necesidad de orquestación de servicios de red, ya que la gestión manual no es suficiente para mantener el ritmo de las rápidas y constantes demandas de los entornos de red modernos.

Es una plataforma de orquestación líder en la industria para redes híbridas. Proporciona una completa automatización del servicio del ciclo de vida para permitirle diseñar y ofrecer servicios de alta calidad de forma más rápida y sencilla.

Primero se verá cómo configurar una instancia de desarrollo de NSO y luego se ejecuta un tutorial básico que demuestra algunas de las características principales:

- Administrar dispositivos
- Configuración de dispositivos
- Servicios

NSO se escala bien tanto hacia arriba como hacia abajo, y se puede ejecutar en una computadora portátil o en una máquina virtual pequeña para fines de laboratorio y desarrollo. Esto nos permite ejecutar esta introducción en un contenedor Docker. Esto

significa que toda la funcionalidad de NSO y la mayoría de los comandos habituales de UNIX están disponibles.[1]

1.1.5 Yang modelado de datos.

El lenguaje YANG de modelado es un estándar desarrollado por el Grupo de Trabajo de Ingeniería de Internet (IETF) para describir la estructura de datos utilizada en la gestión de redes y la configuración de dispositivos de red. YANG se utiliza principalmente en combinación con el Protocolo de Configuración de Red (NETCONF) y también con RESTCONF, que son protocolos de gestión y configuración de dispositivos de red.[11] [12]

El propósito principal de YANG es proporcionar un lenguaje de modelado el cual puede ser entendido por humanos y máquinas, que permita el ingreso de comandos y el intercambio entre dispositivos de red y sistemas de gestión de red, en la Figura 1.3 se observa el modelo YANG.

Algunas características importantes del lenguaje YANG son:[11] [13]

1. Jerarquía de datos: Permite definir una jerarquía de datos estructurados, similar a los árboles de objetos.
2. Tipos de datos: Admite diferentes tipos de datos, como enteros, cadenas, direcciones IP, entre otros, y también permite definir nuevos tipos de datos específicos según las necesidades.
3. Definición de operaciones: Permite definir las operaciones que se pueden realizar en los datos, como lectura, escritura y eliminación.
4. Semántica de datos: YANG permite agregar semántica a los datos, lo que significa que no solo describe la estructura de los datos, sino también su significado y propósito.

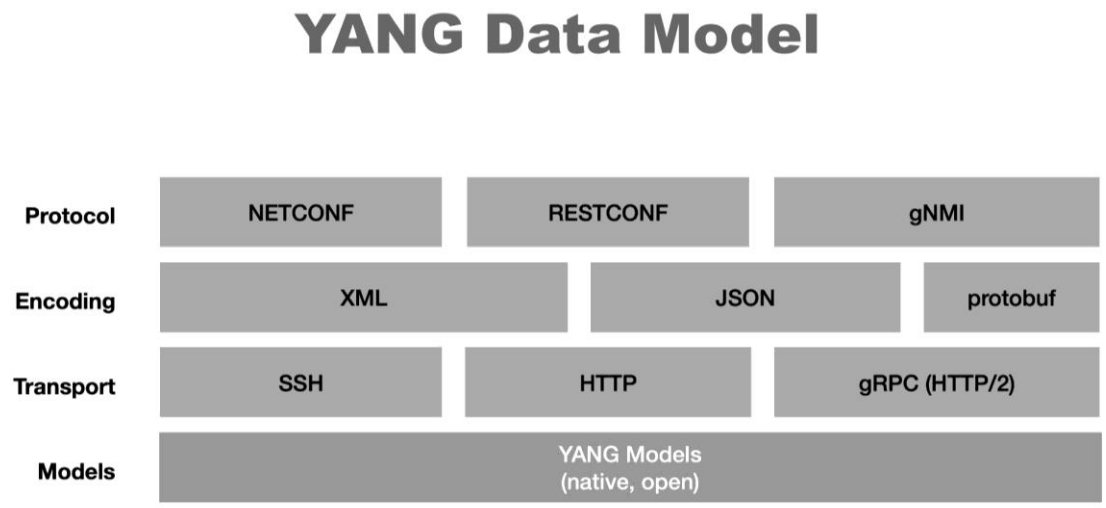

Figura 1.3 Modelo YANG*[14]*

La combinación del lenguaje YANG con los protocolos NETCONF o RESTCONF permite una gestión más eficiente y estandarizada de dispositivos de red, lo que facilita la automatización de tareas y mejora la interoperabilidad entre diferentes equipos y sistemas de gestión.

1.1.6 XML

En el contexto de NSO (Cisco Network Services Orchestrator), XML es uno de los formatos de datos admitidos para la definición de modelos de datos y la manipulación de la configuración y el estado de dispositivos de red. NSO es un software de orquestación de servicios que se utiliza para gestionar y automatizar redes de comunicaciones. Permite a los proveedores de servicios y operadores de redes definir y gestionar modelos de datos que describen la configuración y el estado de los dispositivos de red, así como las operaciones que se pueden realizar en ellos.[15]

XML (Extensible Markup Language) es uno de los formatos de datos que NSO admite para representar estos modelos de datos y también para intercambiar información con los dispositivos de red y los sistemas de gestión. XML proporciona una estructura jerárquica de etiquetas y atributos que permite definir datos estructurados y significativos.[16]

Además de XML, NSO también admite otros formatos de datos como JSON (JavaScript Object Notation) para representar y manipular modelos de datos. La elección entre XML y JSON generalmente depende de las preferencias y requisitos específicos de cada implementación y del sistema o dispositivos con los que NSO interactúe.[15]

1.1.7 Python

Python es un lenguaje de programación muy popular con aplicaciones en diversos campos como la web, desarrollo de software, ciencia de datos y aprendizaje automático (machine learning, ML). Los programadores optan por Python debido a su eficiencia y facilidad de aprendizaje, y es compatible con múltiples plataformas. Además de ser de libre descarga, Python se integra perfectamente con diversos sistemas y acelera el proceso de desarrollo. Su versatilidad y ventajas lo hacen una elección destacada en el mundo de la programación.[17]

Tiene características únicas que lo hacen especial y que son la razón de su amplia adopción:[18]

1. Totalmente gratuito: Python es de código abierto, lo que significa que no tienes que pagar licencias para usarlo.
2. Gran comunidad de apoyo: Debido a su gratuidad, una comunidad activa crea constantemente nuevas bibliotecas y aplicaciones. Casi cualquier pregunta que tengas ya ha sido respondida en foros.
3. Multiparadigma: Combina conceptos de diferentes estilos de programación, haciéndolo flexible y fácil de aprender independientemente de tu nivel de experiencia.
4. Amplitud de aplicaciones: Su naturaleza multiparadigma le permite ser utilizado en diversas áreas, desde el diseño de aplicaciones web hasta la inteligencia artificial y más.
5. Compatibilidad multiplataforma: Puedes ejecutar Python en distintos sistemas operativos, como Windows o Linux, utilizando el intérprete adecuado para cada uno.

API PYTHON NSO

La biblioteca Python de NSO contiene una variedad de API para diferentes propósitos. En Python de NSO se encuentran en dos variantes, las API de bajo nivel y las API de alto nivel.

Las API de bajo nivel son un mapeo directo de las API de NSO C, CDB y MAAPI, las API de alto nivel son una capa de abstracción sobre las API de bajo nivel para que sean más fáciles de usar, para mejorar la legibilidad del código y la tasa de desarrollo para casos de uso comunes. Por ejemplo, devoluciones de llamada de servicios y acciones y secuencias de comandos comunes hacia NSO.[19]

MAAPI- (API de agente de administración) Interfaz orientada hacia el norte que es transaccional y basada en sesiones de usuario. Su función primordial radica en facilitar la lectura de datos de configuración y estados operativos. Además, habilita la capacidad de efectuar tanto la escritura como la confirmación de modificaciones en los datos operativos y de configuración como una única acción coherente. En la Figura 4 se presenta una visualización de la estructura de MAAPi.

MAAPI ofrece la posibilidad de adjuntarse a transacciones en progreso. Esto permite leer modificaciones aún no confirmadas y, en caso necesario, realizar ajustes en estos cambios antes de que se confirmen definitivamente.[19]

2. METODOLOGÍA

2.1. Introducción de uso para NSO

2.1.1 Configuración de dispositivos

Para el desarrollo del laboratorio se utilizar herramientas, comandos propios de CISCO, al igual que otros comandos que nos permitirán emular una red y manejar la automatización, control de las redes.

Se empleará Netsim, una herramienta que posibilita la simulación de redes. A través del comando "ncs-netsim", será viable generar instancias virtuales de dispositivos de red, como enrutadores, switches y cortafuegos. Estos dispositivos se basarán en modelos YANG y estarán configurados para emular diversos escenarios de red. Por el momento, estamos configurando un entorno sencillo que involucra dos dispositivos iOS. La instrucción que utilizamos para lograr esto es el siguiente comando:

Comando: *ncs-netsim create-network cisco-ios-cli-3.8 2 ios*

Ahora se está listos para configurar una instancia de NSO:

Comando: *ncs-setup --dest . --netsim-dir netsim*

- **ncs-setup**: permite preparar la estructura de directorios y archivos necesarios para comenzar a desarrollar servicios y modelos YANG en NSO.
- **--dest .**: El argumento se emplea con el propósito de indicar el directorio de destino donde se establecerá la configuración de los directorios. El punto ("."), en este contexto, denota el directorio actual. Esto implica que la estructura de directorios se generará en la ubicación desde la cual se está ejecutando el comando.

- **--netsim-dir netsim**: El argumento se utiliza para especificar el nombre del directorio donde se creará el entorno de simulación. En este caso, el directorio "netsim", este será creado dentro de un directorio destino que puede ser a gusto del usuario(para la práctica el directorio actual), y dentro de este directorio se configurará la simulación de dispositivos de red.

Puede usar el comando *ls* para observar los archivos creados. Es mejor utilizar ls- l ya que muestra en listado los archivos que se crearon e información detallada, permisos, pesos y fecha de creación como se muestra en la Figura 2.1.

Comando: *ls -l*

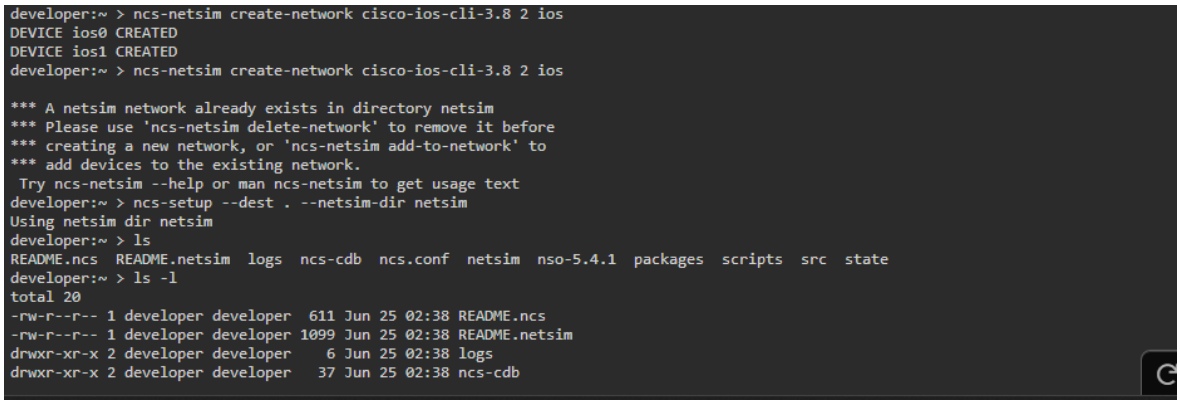

Figura 2.1 Listado de Archivos creados netsim

Se ejecuta el comando "ncs-netsim start", que permite inicializar la simulación, permitiendo la comunicarse entre sí y con otros componentes del entorno de simulación como se muestra en la Figura 2.2. Esto permite a los usuarios que administran la red realizar pruebas y experimentos en un entorno controlado antes de proceder con las configuraciones en redes físicas y en producción.

Comando: *ncs-netsim start*

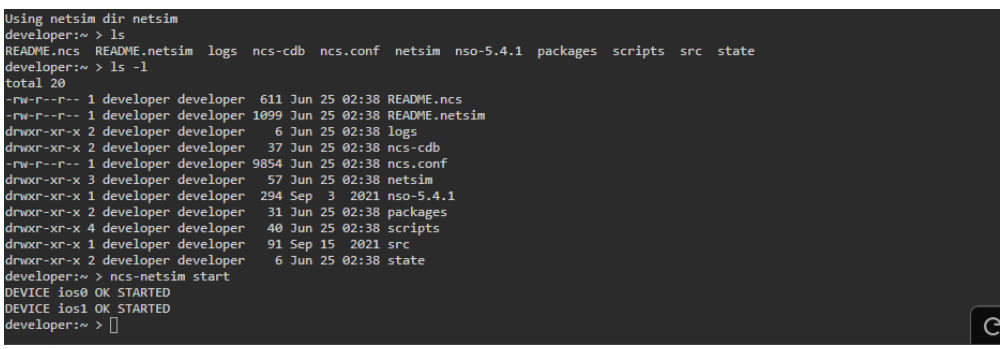

Figura 2.2 Inicio de la simulación de Netsim

Para tener una instancia completa de NSO y en ejecución se ingresa el comando *ncs* el cual se utiliza para interactuar con la plataforma NCS desde la línea de comandos. Al ejecutar el comando *ncs*, este permitirá realizar ciertas tareas:

- Administración de la configuración de la red
- Monitoreo de la red
- Automatización y orquestación
- Generación de informes

Comando: *ncs*

2.1.2 Administrar dispositivos

Se comienza explorando la CLI de NSO y aprender a usarla para administrar dispositivos, para acceder a la interfaz de línea de comandos (CLI) de Cisco Network Control Sistema (NCS) con privilegios de administrador (usuario "admin") y en modo de operación configuración (modo enable), usar el siguiente comando:

Comando: *ncs_cli -u admin -C*

Al ejecutar este comando como se muestra en la Figura 2.4, se inicia la CLI de NCS en la que se pueden ingresar comandos y realizar diversas acciones:

- Administración y configuración de dispositivos de red: Permite gestionar y configurar dispositivos de red a través de la CLI de NCS.
- Monitoreo y solución de problemas de la red: Proporciona herramientas y comandos para monitorear el estado de la red, realizar diagnósticos y solucionar problemas.

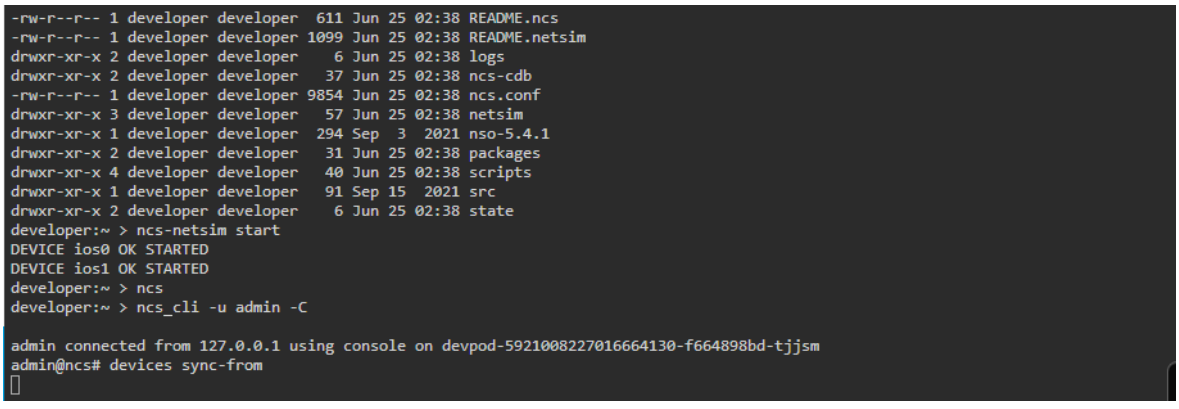

Figura 2.3 Ingreso como administrador en la CLI de NCS

Cuando se ingresa como administrador lo primero a realizar es sincronizar NSO con los dispositivos simulados, para permitir la sincronización de toda la configuración en la red en

la base de datos de configuración (CDB) de NSO. (CDB.- Base de datos de configuración). Se puede realizarlo usando el comando *sync-from*. Esto informará la sincronización exitosa de nuestros dos dispositivos “ios0” y “ios1” como se muestra en la Figura 2.5.

Comando: *devices sync-from*

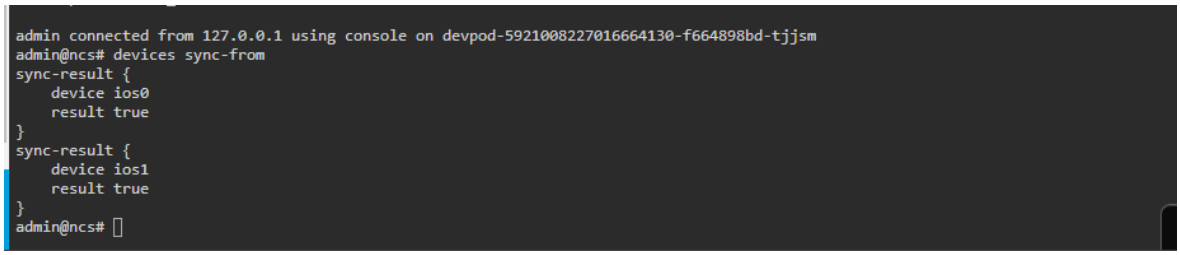

Figura 2.4 Sincronización de los dispositivos

NSO siempre guarda en la CDB un backup de la configuración realizadas en los dispositivos y permite una verificación rápida y fácil de si los dispositivos están en sincronía o no, en la Figura 2.6 se puede observar el resultado de sincronismo del dispositivo.

Comando: *device device ios1 check-sync*

Figura 2.5 Resultado de sincronización dispositivo ios1

2.1.3 Configuración de la red

Ya con la sincronización de los dispositivos se puede ingresar en modo de configuración usando el siguiente comando.

Comando: c*onfig*

Este modo es aquel que nos permitirá examinar y cambiar la configuración de los dispositivos de red, e iniciar una transacción hacia la base de datos.

Para la practica examinar la configuración del dispositivo ios1 usando el comando *show full-configuration*, como se muestra en la Figura 2.7 el comando muestra la información detallada y completa de la configuración del dispositivo, recordar que los dispositivos están bajo la administración de NSO.

Comando: *show full-configuration devices device ios1 config*

```
admin@ncs(config)# show full-configuration devices device ios1 config
devices device ios1
 config
  tailfned device netsim
  tailfned police cirmode
  no service password-encryption
  no cable admission-control preempt priority-voice
  no cable qos permission create
  no cable qos permission update
  no cable qos permission modems
  ip source-route
  no ip cef
  ip vrf my-forward
   bgp next-hop Loopback1
  !
  no ip forward-protocol nd
  no ip http server
  no ip http secure-server
  ip community-list 1 permit
  ip community-list 2 deny
devices device ios1
 config
  tailfned device netsim
  tailfned police cirmode
  no service password-encryption
  no cable admission-control preempt priority-voice
  no cable qos permission create
  no cable qos permission update
  no cable qos permission modems
  ip source-route
```

Figura 2.6 Vista completa del dispositivo ios1

La CLI (Interfaz de línea de Comandos) de NSO tiene un comportamiento similar a la mayoría de los CLI de algunos sistemas, por lo tanto para avanzar y ver la información completa hasta la finalización del comando ingresado se lo puede realizar presionando ? o TAB.

Dentro del modo de configuración, se puede realizar cambios fácilmente de configuración dentro de un dispositivo.

Para poder realizar estos cambios dentro del contexto primero se debe apuntar al dispositivo que se quiere modificar como se muestra en la Figura 2.8 y se ingresa el comando siguiente.

Comando: *devices device ios0 config*

```
Entering configuration mode terminal
admin@ncs(config)# show full-configuration devices device ios1 config | nomore
devices device ios1
 config
  no service password-encryption
  no cable admission-control preempt priority-voice
  no cable qos permission create
  no cable qos permission update
  no cable qos permission modems
  ip source-route
  no ip cef
  no ip forward-protocol nd
  no ipv6 source-route
  no ipv6 cef
 !
!
admin@ncs(config)# show full-configuration devices device ios1 config | include bgp
admin@ncs(config)# show full-configuration devices device ios1 config | include bgp
admin@ncs(config)# devices device ios0 config
admin@ncs(config-config)#
```

Figura 2.7 Configuración del dispositivo ios0

Si se ingresa el comando *full-configuration* en este contexto se verá solo la configuración del dispositivo “ios0”, nos ayudamos de mejor manera con el comando *nomore* que permite no tener ninguna interrupción en la visualización.

Comando: *show full-configuration | nomore*

Se realiza el cambio de nombre de host:

Comando: *ios:hostname nso.cisco.com*

El cambio es inmediatamente en la configuración del dispositivo, pero los cambios no son visibles para el usuario; para poder observar dichos cambios se agregarán algunos comandos más para continuar y verificar que los cambios si se realizaron. Para poder observar los cambios realizados en los dispositivos debemos regresar a un contexto superior, por lo tanto, si se desea volver al contexto de nivel superior (config), se ingresa el comando *top.*

Comando: *top*

Para poder observar el cambio del hostname realizado en el dispositivo ios0 se ingresa el siguiente comando, la vista q valida que en realidad se realizó el cambio se muestra en la Figura 2.9.

Comando: *show configuration*

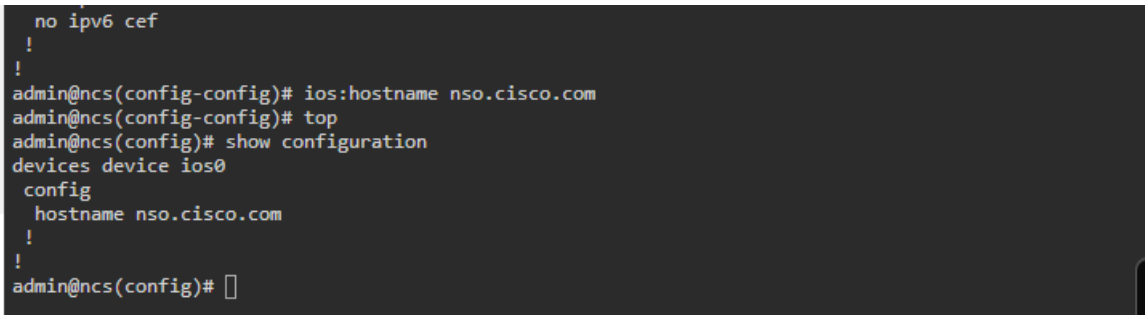

Figura 2.8 Hostname dispositivo ios0

Se puede realizar cambios a varios dispositivos simultáneamente, por ejemplo se puede realizar la configuración de una contraseña para todos los dispositivos, para ello se usará el siguiente comando *enable password magic*. La Figura 2.10 nos permitirá visualizar dicho cambio a los dispositivos.

Comando: *devices device * config ios:enable password magic*

Dentro del comando se observa que se ingresa (*), este símbolo representa que los cambios que se vayan a realizar lo generen a todos los dispositivos, y con “enable” se informa que tome el cambio sobre la información antigua para un almacenamiento de la contraseña de una manera más segura utilizando un cifrado más fuerte.

Se verifica los cambios realizados nuevamente con el comando.

Comando: *show configuration*

```
admin@ncs(config)# devices device * config ios:enable password magic
admin@ncs(config-config)# show configuration
devices device ios0
 config
  hostname nso.cisco.com
  enable password magic
 !
!
devices device ios1
 config
  enable password magic
 !
!
admin@ncs(config-config)#
```

Figura 2.9 Cambio de contraseña en los dispositivos ios

Después de realizar alguna configuración a uno o varios dispositivos, se tiene dos comandos principales que se pueden utilizar que son: *revert* (descarta los cambios realizados en los dispositivos) o commit (permite guardar los cambios realizados en los dispositivos).

Al realizar configuraciones en los dispositivos es recomendable usar el comando *dry-run*, el cual, permitirá identificar posibles problemas, errores o resultados no deseados antes de comprometerse a realizar la acción de guardado y almacenar los cambios de manera definitiva, como se puede observar en la Figura 2.11.

Comando: *commit dry-run outformat native*

```
admin@ncs(config-config)# commit dry-run outformat native
native {
    device {
        name ios0
        data hostname nso.cisco.com
             enable password magic
    }
    device {
        name ios1
        data enable password magic
    }
}
admin@ncs(config-config)#
```

Figura 2.10 Verificación de cambios comando dry-run

Después de realizar la verificación de que los cambios realizados son los correctos se procede a guardar los cambios realizados usando el comando *commit*, este nos informara mediante mensaje de que los cambios en las configuraciones han sido realizados. Una vez terminado todo el proceso se usa igual el comando *end* para finalizar las acciones.

Comando: *commit*

Comando: *end*

2.1.4 Servicios básicos

Los servicios en NSO permiten realizar una simplificación en las configuraciones de los dispositivos, mejorando la automatización y el aprovisionamiento de servicios.

Para agregar un servicio se debe crear un paquete de servicio, que tendrá algunos archivos y directorios ya predefinidos. En esta práctica de laboratorio se copiarán algunos archivos predefinidos para poder comenzar con un servicio simple.

Para agregará un paquete de servicio y la estructura del paquete utilizando el comando ncs-*make*-package, pero para realizar todo este proceso hay que hacerlo desde el “Bash” es decir fuera del contexto de configuración.

Si se encuentra aún en la parte de configuración de Red salimos para regresar al bash ingresamos el comando *Exit*. Una vez que se encuentre en el bash ingresar el siguiente comando.

Comando: *ncs-make-package --service-skeleton template --dest ~/src/packages/simple-service simple-service*

- **ncs-make-package**: Este es el comando principal para crear un paquete de servicio en NSO.
- **--service-skeleton template**: Indica que deseas utilizar una plantilla base con nombre "template" como base para el paquete de servicio
- **--dest ~/src/packages/simple-service**: Establece la ubicación de destino donde se generará la estructura del paquete de servicio.
- **simple-s**: Es un argumento que se utiliza para proporcionar un nombre o identificador para el paquete de servicio

Las siguientes líneas es la estructura de un paquete simple de servicio para uso en NSO. Permite establecer la contraseña de habilitación en un dispositivo Cisco IOS.

Estructura de Código:

```
printf 'module simple-service {
  namespace "http://com/example/simpleservice";
  prefix simple-service;
  import tailf-ncs { prefix ncs; }
  list simple-service {
    key name;
    uses ncs:service-data;
    ncs:servicepoint simple-service;

    leaf name {
      type string;
    }
    leaf device {
      type leafref {
        path "/ncs:devices/ncs:device/ncs:name";
      }
    }
    leaf secret {
      type string;
    }
  }
}
' > ~/src/packages/simple-service/src/yang/simple-service.yang
 printf    '<config-template    xmlns="http://tail-f.com/ns/config/1.0"    servicepoint="simple-
Service”>
  <device xmlns=http://tail-f.com/ns/ncs>
  <device>
     <name>{./device}</name>
```

```
    <config>
      <enable xmlns="urn:ios">
        <password>
          <secret>{./secret}</secret>
        </password>
      </enable>
    </config>
  </device>
 </devices>
</config-template>
' > ~/src/packages/simple-service/templates/simple-service-template.xml
```

Con el siguiente comando se inicializa el proceso de construcción del paquete según la estructura especificada

Comando: *make -C ~/src/packages/simple-service/src*

Si se quiere ver todo el proceso y constatar que el servicio se completó correctamente, se coloca el siguiente comando. La Figura 2.12 muestra en el proceso de creación así como la confirmación de un completado exitoso.

Comando: *echo "packages reload"* | *ncs_cli -C -u admin*

```
service-ann.yang"` \
        --fail-on-warnings \
         \
        -c -o ../load-dir/simple-service.fxs yang/simple-service.yang
make: Leaving directory '/home/developer/src/packages/simple-service/src'
developer:src > echo "packages reload" | ncs_cli -C -u admin

>>> System upgrade is starting.
>>> Sessions in configure mode must exit to operational mode.
>>> No configuration changes can be performed until upgrade has completed.
>>> System upgrade has completed successfully.
reload-result {
    package cisco-ios-cli-3.8
    result true
}
reload-result {
    package simple-service
    result true
}
developer:src >
```

Figura 2.11 Verificación de la creación del paquete comando echo

Regresar al CLI de NSO, para verificar el estado de instalación del paquete, en la Figura 2.13 se puede observar el estado del paquete.

Comando: *ncs_cli -C -u admin*

Comando: *show packages package oper-status*

```
admin@ncs# show packages package oper-status
packages package cisco-ios-cli-3.8
 oper-status up
packages package simple-service
 oper-status up
admin@ncs#
```

Figura 2.12 Estado del paquete instalado

Si se requiere tener una instancia de servicio se debe ingresar al modo de configuración (config) y crear una instancia de servicio que permitirá realizar ciertas configuraciones.

Comando: *config*

Proceder con la creación de una instancia de servicio, esta tendrá como nombre “test1” que y será configurada en el dispositivo “ios0” la seguridad es importante por lo tanto tendrá una contraseña de nombre “mypasswd”, para realizar todo este proceso se ingresa el siguiente comando.

Comando: *Simple-service test1 device ios0 secret mypasswd*

Recordar que siempre se debe realizar la respectiva verificación antes de realizar cualquier cambio, así que, usar *show configuration* para ver la información que posee “test1” en relaciona a los cambios a realizarse en el dispositivo “ios0”, en la Figura 2.14 se observa la información.

Comando: *show configuration*

```
admin@ncs(config-simple-service-test1)# show configuration
simple-service test1
 device ios0
 secret mypasswd
!
```

Figura 2.13 Resultado de los datos en test1

Para visualizar la creación del servicio y el cambio en el dispositivo “ios0”. De forma general sobre todo el proceso de cabio realizado se usa el comando *dry-run*, la Figura 2.15 muestra la información desplegada en relación con el dispositivo “ios0”.

Comando: *commit dry-run*

Figura 2.14 Vista del proceso de cambio del dispositivo ios0

La visualización también puede generarse de forma nativa, es decir que muestre lo que se configuraría en los dispositivos, la Figura 2.16 nos permite visualizar la información nativa del dispositivo “ios0”.

Comando: *commit dry-run outformat native*

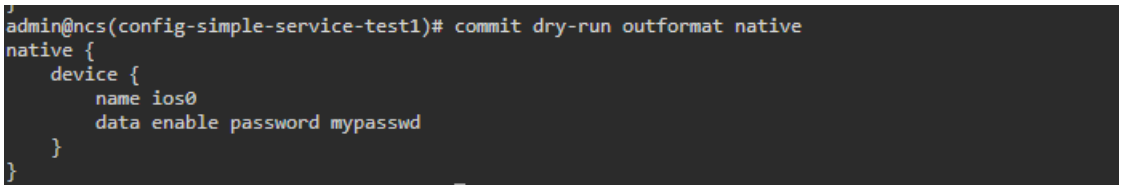

Figura 2.15 Vista del cambio en el dispositivo ios0 nativa

Se procede a guardar los cambios realizados usando el comando Commit

Comando: *commit*

Si se comete un error, por ejemplo, en la contraseña y esta debe ser “securepasswd” en su lugar de la actual generada. Se puede realizar el cambio del servicio con el siguiente comando.

Comando: *secret securepasswd*

Se verifica el cambio con el comando dry-run, si el cambio es correcto se procede a guardar la nueva contraseña con el comando *commit*, si se requiere regresar al anterior contexto superior lo realizamos con el comando *top*.

Comando: *commit dry-run*

Comando: *commit*

Comando: *top*

Existen comandos que permiten ingresar a varias acciones y ser ejecutar sobre un servicio. El primero de ellos es *get-modifications*, que muestra cuál fue el resultado de este servicio, como se muestra en la Figura 2.17, que muestra toda la información que ha sido agregada al dispositivo “ios0”.

Comando: *simple-service test1 get-modifications*

```
admin@ncs(config)# simple-service test1 get-modifications
cli {
    local-node {
        data  devices {
                    device ios0 {
                        config {
                            enable {
                                password {
              -                         secret magic;
              +                         secret securepasswd;
                                }
                            }
                        }
                    }
                }

    }
}
admin@ncs(config)#
```

Figura 2.16 Verificación del resultad del servicio Test1

Otra acción que se puede realizar es eliminar el servicio y eso se puede ejecutar con el siguiente comando.

Comando: *no simple-service test1*

2.2. Introducción a cisco network services orchestrator

NSO CISCO es una aplicación de Linux que organiza el ciclo de vida de configuración de los dispositivos de red físicos y virtuales, permitiendo recopilar, analizar y almacenar el estado o los estados de configuración de todos los dispositivos de red, las cuales son administradas en una base de datos de configuración llamada (CDB). Esto permite a que los usuarios y las aplicaciones pueden pedir a NSO que crea, lea, actualice o elimine la configuración de manera programática de cualquier dispositivo dentro de una red. Además NSO utiliza paquetes de software llamados Network Element Drivers (NED) para facilitar las interacciones de telnet, SSH o API hacia los dispositivos que administra.

2.2.1 CONFIGURACIÓN DE NSO

Este laboratorio de aprendizaje se centrará en la instalación local de NSO (servidor CISCO), ya que es más fácil navegar y ejecutar los comando y configuraciones de cierto

modo localmente en una computadora portátil, esta comunicación se lo realizara mediante el uso de una VPN para el caso práctico la VPN de Cisco.

La topología de configuración se presenta en la Figura 2.18 y la vista de datos de cada servidor está en la Tabla 2.1.

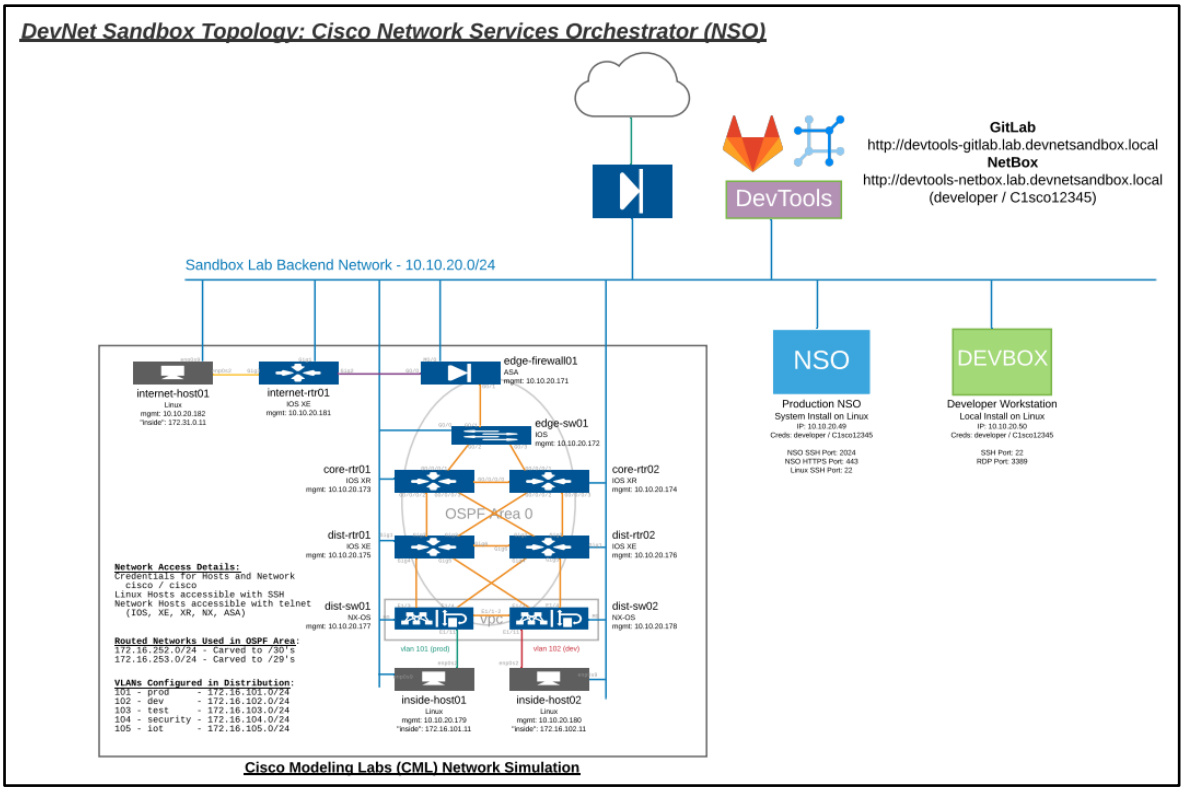

Figura 2.17 Topología de red para NSO*[20]*

Información principal de los dispositivos.

- **Anfitrión NSO/NCS**
    - Dirección: 10.10.20.49
    - Nombre de usuario: desarrollador
    - Contraseña: C1sco12345
    - Puerto SSH para host Linux: 22
    - Puerto HTTPS para NSO GUI/API: 443
      Conexión de ejemplo: https://10.10.20.49
    - Puerto SSH para acceso directo a NSO: 2024
      Conexión de ejemplo: ssh -p 2024 desarrollador@10.10.20.49

- **caja de desarrollo**
    - Dirección: 10.10.20.50
    - Nombre de usuario: desarrollador
    - Contraseña: C1sco12345
    - Puerto SSH: 22
    - Puerto RDP: 3389
- **Herramientas NetDevOps**
    - **GitLab** : control de código fuente y CICD
        - Dirección: http://devtools-gitlab.lab.devnetsandbox.local
        - Nombre de usuario: desarrollador
        - Contraseña: C1sco12345
    - **NetBox** : Fuente de la verdad: IPAM y DCIM
        - Dirección: http://devtools-netbox.lab.devnetsandbox.local
        - Nombre de usuario: desarrollador
        - Contraseña: C1sco12345

También se tiene una tabla con toda la información de los dispositivos de red el cual se posee toda la información de red.

Tabla 2.1 Información de red de los dispositivos del Laboratorio [20]

Componente	Tipo	Dirección IP	Método	Cartas credenciales
internet-host01	linux	10.10.20.182	SSH	cisco / cisco
internet-rtr01	IOS XE	10.10.20.181	TELNET	cisco / cisco
cortafuegos de borde01	COMO UN	10.10.20.171	TELNET	cisco / cisco
borde-sw01	iOS	10.10.20.172	TELNET	cisco / cisco
core-rtr01	IOS XR	10.10.20.173	TELNET	cisco / cisco
core-rtr02	IOS XR	10.10.20.174	TELNET	cisco / cisco
dist-rtr01	IOS XE	10.10.20.175	TELNET	cisco / cisco
dist-rtr02	IOS XE	10.10.20.176	TELNET	cisco / cisco
dist-sw01	NX-OS	10.10.20.177	TELNET	cisco / cisco
dist-sw02	NX-OS	10.10.20.178	TELNET	cisco / cisco
dentro-host01	linux	10.10.20.179	SSH	cisco / cisco
dentro-host02	linux	10.10.20.180	SSH	cisco / cisco
caja de red	devtools-netbox.lab.devnetsandbox.local			desarrollador / C1sco12345
GitLab	devtools-gitlab.lab.devnetsandbox.local			desarrollador / C1sco12345

Primero se realiza la conexión por la VPN de Cisco para proceder con las configuraciones de NSO y los diferentes dispositivos de la red simulada. Una vez conectado en la VPN se inicia sesión en DevBox en el Laboratorio reservable de NSO, como se observa en la Figura 2.19.

Comando: *ssh developer@10.10.20.50*

```
$ ssh developer@10.10.20.50
Warning: Permanently added '10.10.20.50' (ECDSA) to the list of known hosts
developer@10.10.20.50s password:
Last login: Thu Mar 19 11:56:22 2020 from 10.20.0.16
(py3venv) [developer@devbox ~]$
```

Figura 2.18 inicio de sesión por VPN a caja de desarrollo

Existe una carpeta nombrada nso en el directorio de inicio del usuario, que se creó al desempaquetar el instalador de NSO, la Figura 2.20 muestra el listado de lo que contiene este directorio.

Comando: *ls –l nso-5.4.1/*

```
developer:~ > ls -l nso-5.4.1/
total 488
-rw-r--r-- 1 developer developer 371246 Sep 14  2020 CHANGES
-rw-r--r-- 1 developer developer  94792 Sep 14  2020 LICENSE
-rw-r--r-- 1 developer developer   7155 Sep 14  2020 README
-rw-r--r-- 1 developer developer    375 Sep  3  2021 VERSION
drwxr-xr-x 1 developer developer   4096 Sep 14  2020 bin
drwxr-xr-x 1 developer developer     58 Sep 14  2020 doc
drwxr-xr-x 1 developer developer     34 Sep 14  2020 erlang
drwxr-xr-x 1 developer developer     17 Sep 14  2020 etc
drwxr-xr-x 1 developer developer    177 Sep 14  2020 examples.ncs
drwxr-xr-x 1 developer developer    184 Sep 14  2020 include
drwxr-xr-x 1 developer developer     17 Sep  3  2021 java
drwxr-xr-x 1 developer developer    112 Sep 14  2020 lib
drwxr-xr-x 1 developer developer     54 Sep 14  2020 man
```

Figura 2.19 Paquete nso preinstalado

Mirar el contenido del nso/packages/neds para ver la información que posee, la Figura 2.21 muestra los archivos dentro del directorio neds.

Comando: *ls nso-5.4.1/packages/neds/*

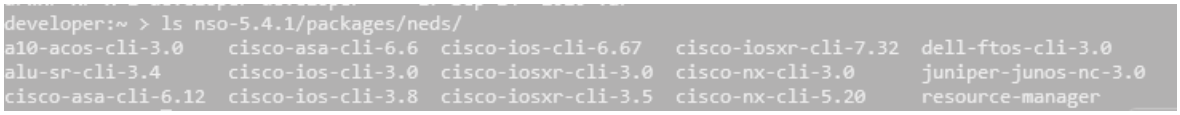

Figura 2.20 Paquete NED

En el capítulo anterior se revisó el manejo del comando **ncs-setup** lo que permite es la creación de instancias de NSO dentro del servidor de laboratorio.

Se puede realizar consultas y el sudo de complementos al comando por ejemplo el complemento --help, para el caos practico de este laboratorio solo se usará dos comandos de importancia para el desarrollo de NSO las cuales son:

--dest. - Define el directorio donde desea configurar NSO (si el directorio no existe, el sistema lo crea).

--package. - Define los NED que desea que tenga instalada esta instancia de NSO. Puede especificar esta opción varias veces.

Después de realizar el desempaquetado y verificada la información de NSO, se debe obtener el archivo “ncsrc” por lo tanto nos dirigimos al directorio de instalación de NSO donde se encuentra el archivo y ejecutamos el siguiente comando. Lo que realiza “source” es cargar y ejecutar el archivo “ncsrc”.

Comando: *source ~/nso/ncsrc*

Ejecute los siguientes comandos para configurar una instancia de NSO en el directorio actual con los NED de IOS, NX-OS, IOS-XR y ASA, la Figura 2.22 muestra el proceso de configuración.

Comando: *ncs-setup* \

Comando: *--package nso/packages/neds/cisco-nx-cli-5.23* \

Comando: *--package nso/packages/neds/cisco-asa-cli-6.6* \

Comando: *--package nso/packages/neds/cisco-ios-cli-6.91* \

Comando: *--package nso/packages/neds/cisco-iosxr-cli-7.45* \

Comando: *--dest nso-instance*

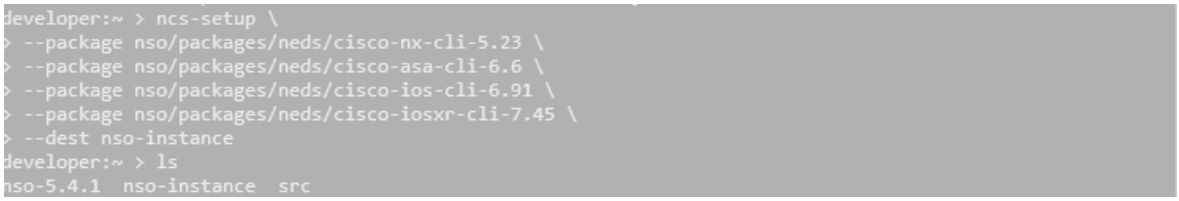

Figura 2.21 Configuración de una instancia nso-intance

Como se puede observar el directorio “nso-instance” encontrará varios archivos y carpetas nuevos. Aquí hay un par de archivos y carpetas notables, la Figura 2.23 muestra la información que posee el directorio “nso-instance”.

- ncs.conf: archivo de configuración de la aplicación NSO. Se utiliza para personalizar aspectos de la instancia de NSO (cambiar puertos, habilitar o deshabilitar funciones, etc.).
- packages/: directorio que tiene enlaces simbólicos a los NED a los que se hace referencia a los argumentos –package de la configuración.
- logs/: Directorio que contiene todos los registros de NSO. Utilice este directorio para verificar errores y solucionar problemas.

Comando: *ls nso-instance/*

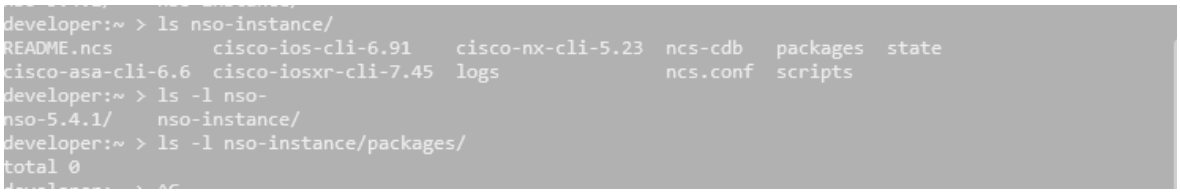

Figura 2.22 Vista interna del directorio nso-instance

Pata inicializar la instancia de NSO, se ingresar al directorio nso-instance con el comando cd + (directorio) y escribir el comando *ncs*. El proceso de carga y ejecución del comando tarda unos segundos.

Comando: *cd nso-instance/*

Comando: *ncs*

Para verificar que NSO se está ejecutando se usa el comando *ncs --status* | *grep status*. la Figura 2.24 muestra el estado de inicialización de “ncs”.

Comando: *ncs --status* | *grep status*

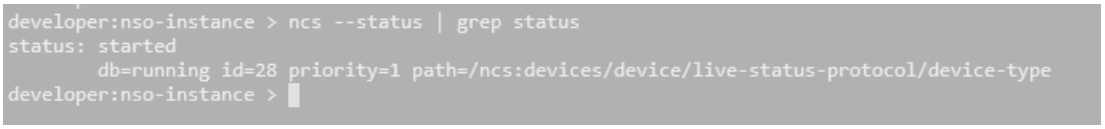

Figura 2.23 Verificación de que esta activo ncs

2.2.2 Llenando de la instancia de nso

Ahora que está la instancia de NSO creada y en ejecución, se requiere algunas cosas antes de poder comenzar a automatizar:

- authgroup habilita la autenticación de credenciales del dispositivo.
- Información del dispositivo para completar la lista de dispositivos.
- Los grupos de dispositivos son útiles, pero no obligatorios.

Ingresar a la instancia de NSO con el comando *ncs_cli -C -u admin*.

Comando: *ncs_cli -C -u admin.*

Ingresar al "modo de configuración" con el comando *config*.

Comando: *config*

Para la configuración inicial se va a tomar en consideración un nuevo authgroup llamado “labadmin”. Este grupo utilizará una combinación predeterminada de nombre de usuario y contraseña (cisco/cisco), además una contraseña secundaria “cisco”, la Figura 2.25 muestra el proceso de agrupación y configuración de contraseñas tanto primaria como secundaria. Utilice los siguientes comandos:

Comando: *devices authgroups group labadmin*

Comando: *default-map remote-name cisco*

Comando: *default-map remote-password cisco*

Comando: *default-map remote-secondary-password cisco*

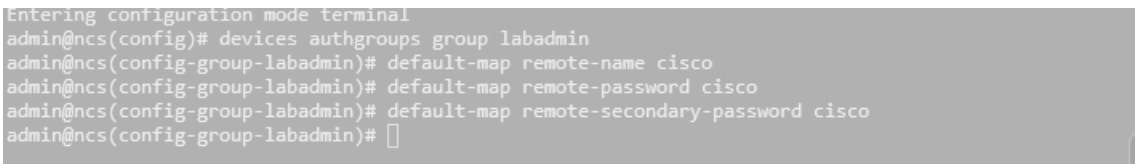

Figura 2.24 Configuración del authogroup, definición de usuario y contraseña

Regresar a un contexto superior con el comando *top* con esto regresamos al contexto “config”, ingresamos el comando *show configuration* para que muestre la información ingresada este comando ya se lo había visto previamente en la introducción de uso, la Figura 2.26 muestra la información del grupo creado.

Comando: top

Comando: show configuration

```
admin@ncs(config-group-labadmin)# top
admin@ncs(config)# show configuration
devices authgroups group labadmin
 default-map remote-name   cisco
 default-map remote-password $9$bBBM/QtfOaPt3/9j5XXdb3pf19s4tI7HmW0vizxvolw=
 default-map remote-secondary-password $9$NWVkh+MuU0VJIwMKts8b8L/YzAcZtnG8KGs2S8uz38w=
!
admin@ncs(config)#
```

Figura 2.25 Vista previa de los datos ingresados del grupo antes de ser guardados

Se guarda la configuración después de la verificación con el comando *commit*

Comando: *commit*

Se agregará un dispositivo al inventario del grupo “labadmin”. Para agregar un dispositivo se necesita la siguiente información:

- La dirección IP o FQDN del dispositivo.
- El protocolo (SSH, Telnet, HTTP, REST, etc.) y el puerto (si no es estándar) para conectarse al dispositivo.
- El authgroup que se usará para el dispositivo (que debe confirmarse antes de agregar dispositivos).
- El NED (controlador de dispositivo) que se utilizará para conectarse al dispositivo.

En este laboratorio se agregará el primer dispositivo paso a paso y luego se agregará el resto de forma masiva.

En el paso anterior se quedó en la ruta (config), se mantendrá en esta dirección y se ingresará el siguiente comando que permitirá, ingresar un dispositivo, la Figura 2.27 muestra el proceso de agrupación de un dispositivo.

Comando: *devices device edge-sw01*

Comando: *address 10.10.20.172*

Comando: *authgroup labadmin*

Comando: *device-type cli ned-id cisco-ios-cli-6.91*

Comando: *device-type cli protocol telnet*

Comando: *ssh host-key-verification none*

Comando: *commit*

```
Entering configuration mode terminal
admin@ncs(config)# devices device edge-sw01
admin@ncs(config-device-edge-sw01)# address 10.10.20.172
admin@ncs(config-device-edge-sw01)# authgroup labadmin
admin@ncs(config-device-edge-sw01)# device-type cli ned-id cisco-ios-cli-6.91
--------------------------------------------------------------------------------^
```

Figura 2.26 Configuración del dispositivo 1 IP 10.10.20.172

Al ingresar varios comandos de configuración es muy común perderse y no conocer la ruta en la que estamos configurando, para conocer la ruta se ingresa el comando *pwd,* Al ejecutar este comando, se visualiza la ruta completa del directorio en el que estás ubicado en ese momento*.* La Figura 2.28 muestra los resultados obtenidos.

Comando: *pwd*

```
admin@ncs(config-device-edge-sw01)# pwd
Current submode path:
devices device edge-sw01
admin@ncs(config-device-edge-sw01)#
```

Figura 2.27 verificación en el contexto CLI

Utilice el comando *connect* para determinar si puede conectarse al dispositivo con NSO, el resultado muestra que actualmente está locked (bloqueado). El modo predeterminado de NSO para dispositivos es un estado bloqueado que evita que NSO manipule un dispositivo antes de que el administrador de la red esté listo.

Comando: *connect*

Para Desbloquear el dispositivo se usa el comando *admin-state* y *commit* para realizar el guardado de la configuración, por lo tanto ingresar el siguiente comando.

Comando: *state admin-state unlocked*

Comando: *commit*

Ahora se puede repetir el proceso para los dispositivos restantes, y hay algunas opciones.

- Puede escribir manualmente los comandos para cada dispositivo: una gran práctica, pero un poco aburrido.

- Puede usar una API para agregar dispositivos a la lista de dispositivos.
- Puede consultar la instalación de un servidor NSO esto dentro del sistema de producción del laboratorio. Al ingresar este ya posee una lista completa de dispositivos y guardada en un archivo.

En este laboratorio se utilizará la tercera opción por lo tanto se requiere abrir una nueva terminal del MobaXterm para acceder al servidor de NSO instalada en el Laboratorio con las credenciales de SSH proporcionadas en las instrucciones e inicie sesión en la CLI de la NSO.

Comando: *ssh 10.10.20.49*

Comando: *password: C1sco12345*

La Figura 2.29 muestra la conexión que se realiza con la dirección IP 10.10.20.49.

```
Warning: Permanently added 10.10.20.49 (ECDSA) to the list of known hosts.
developer@10.10.20.49s password:
```

Figura 2.28 Ingreso por ssh a **Anfitrión NSO/NCS**

Ejecute el comando *show running-config devices device* | *de-select config* para obtener toda la configuración almacenada en NSO, la Figura 2.30 muestra toda la información y configuraciones de los dispositivos.

Comando: *show running-config devices device* | *de-select config*

```
admin@ncs# show running-config devices device | de-select config
devices device core-rtr01
address   10.10.20.173
ssh host-key-verification none
authgroup labadmin
device-type cli ned-id cisco-iosxr-cli-7.45
device-type cli protocol telnet
state admin-state unlocked
!
devices device core-rtr02
address   10.10.20.174
ssh host-key-verification none
authgroup labadmin
device-type cli ned-id cisco-iosxr-cli-7.45
device-type cli protocol telnet
state admin-state unlocked
!
devices device dist-rtr01
address   10.10.20.175
ssh host-key-verification none
authgroup labadmin
device-type cli ned-id cisco-ios-cli-6.91
device-type cli protocol telnet
state admin-state unlocked
!
devices device dist-rtr02
address   10.10.20.176
ssh host-key-verification none
authgroup labadmin
device-type cli ned-id cisco-ios-cli-6.91
device-type cli protocol telnet
state admin-state unlocked
```

Figura 2.29 Vista de todos los dispositivos configurados

Se copia el resultado de ese comando y se guarda en un archivo llamado “nso-device-config.txt” en el directorio de inicio. Puede usar editores de texto vim, nano o cat.

Cierre la sesión del terminal con la IP 10.10.20.49 para proceder a instalar en el sistema de producción la cual tiene la dirección IP 10.10.20.50. Ingrese al modo de configuración de la CLI de NSO.

Ingrese al modo config y cargue el archivo con el comando *load merge*, la Figura 2.31 muestra el proceso de carga con el comando *load merge*.

Comando: *ncs_cli -C -u admin*

Comando: *conf*

Comando: *load merge /home/developer/nso-device-config.txt*

```
admin@ncs# conf
Entering configuration mode terminal
admin@ncs(config)# load merge /home/developer/nso-device-config.txt
Loading.
1.95 KiB parsed in 0.95 sec (2.05 KiB/sec)
```

Figura 2.30 Carga de los dispositivos masivamente

Ahora use el comando *commit* para agregar y guardar estos dispositivos a NSO. Use también el comando *end* para salir del modo config.

Comando: *commit*

Comando: *end*

Verifique que los dispositivos se hayan agregado a NSO con *show devices list*, la Figura 2.32 muestra el listado de los dispositivos agregados.

Comando: *show devices list.*

```
admin@ncs# show devices list
NAME             ADDRESS       DESCRIPTION  NED ID                ADMIN ST
-------------------------------------------------------------------------
core-rtr01       10.10.20.173  -            cisco-iosxr-cli-7.45  unlocked
core-rtr02       10.10.20.174  -            cisco-iosxr-cli-7.45  unlocked
dist-rtr01       10.10.20.175  -            cisco-ios-cli-6.91    unlocked
dist-rtr02       10.10.20.176  -            cisco-ios-cli-6.91    unlocked
dist-sw01        10.10.20.177  -            cisco-nx-cli-5.23     unlocked
dist-sw02        10.10.20.178  -            cisco-nx-cli-5.23     unlocked
edge-firewall01  10.10.20.171  -            cisco-asa-cli-6.6     unlocked
edge-sw01        10.10.20.172  -            cisco-ios-cli-6.91    unlocked
internet-rtr01   10.10.20.181  -            cisco-ios-cli-6.91    unlocked
admin@ncs#
```

Figura 2.31 Vista de los dispositivos configurados

Se verifica que NSO pueda comunicarse con estos dispositivos intentando conectarse a todos ellos con el comando *devices connect*.

Aprendizaje Estado actual de la red

Aunque NSO logra establecer conexión con la red, carece de conocimiento acerca de la configuración de red en los dispositivos en uso. Este ejercicio tiene como objetivo instruir sobre la configuración de la red.

Se verifica que no haya una configuración a nivel de dispositivo dentro de NSO para ello se usa el comando *show*, la Figura 2.33 muestra la configuración de red del dispositivo edge-sw01.

Comando: *show running-config devices device edge-sw01 config*

```
admin@ncs# show running-config devices device edge-sw01 config
devices device edge-sw01
config
no service password-encryption
no cable admission-control preempt priority-voice
no cable qos permission create
no cable qos permission update
no cable qos permission modems
ip source-route
no ip cef
no ip forward-protocol nd
no ipv6 cef
no dot11 syslog
!
!
admin@ncs#
```

Figura 2.32 Vista de la configuración de un dispositivo

El comando *sync-from* permite que los dispositivos de red aprendan la configuración actual, La Figura 2.34 muestra el estado obtenido de la sincronización en cada uno de los dispositivos.

Comando: *devices sync-from*

```
admin@ncs# devices sync-from
sync-result {
    device core-rtr01
    result true
}
sync-result {
    device core-rtr02
    result true
}
sync-result {
    device dist-rtr01
    result true
}
```

Figura 2.33 Sincronización de los dispositivos

Verificar con el comando *running-configuration* que los cambios de red en la NSO se hayan efectuado a todos los dispositivos, Se realizara este proceso al dispositivo edge-sw01, se observa en la Figura 2.35 que en relación con la primera consulta de la Figura 2.34 existe un cambio.

Comando: *show running-config devices device edge-sw01 config*

```
admin@ncs# show running-config devices device edge-sw01 config
devices device edge-sw01
 config
  hostname  edge-sw01
  tailfned police cirmode
  version   15.2
  service timestamps debug datetime msec
  service timestamps log datetime msec
  no service password-encryption
  service compress-config
  no cdp run
  vrf definition Mgmt-intf
   address-family ipv4
    exit-address-family
   !
   address-family ipv6
    exit-address-family
   !
  !
  enable password cisco
  no cable admission-control preempt priority-voice
  no cable qos permission create
  no cable qos permission update
  no cable qos permission modems
  ip source-route
  ip cef
  no ip domain lookup
  ip ssh server algorithm encryption aes128-ctr aes192-ctr aes256-ctr
  ip ssh client algorithm encryption aes128-ctr aes192-ctr aes256-ctr
  ip forward-protocol nd
  no ip http server
  no ip http secure-server
  ip route vrf Mgmt-intf 0.0.0.0 0.0.0.0 10.10.20.254
  no ipv6 cef
```

Figura 2.34 Vista de la configuración después de la sincronización

Agrupación de dispositivos

Este paso simplifica el trabajo con la red. Se puede crear grupos de dispositivos para organizarlos en grupos lógicos, esto para aplicar una configuración similar o realizar acciones similares a un grupo. En el laboratorio se crean los siguientes grupos definidos:

Grupos basados en sistemas operativos de red

- IOS-DEVICES. - Es un dispositivo electrónico que se ejecuta en **iOS**
- XR-DEVICES. - Son dispositivos que combinan tecnologías inmersivas de realidad virtual (VR), realidad aumentada (AR) y realidad mixta (MR)
- NXOS-DEVICES. - Son switches de la serie Nexus con un sistema operativo desarrollado por Cisco System.

- ASA-DEVICES. – Son dispositivos de seguridad de red desarrollados por Cisco Systems.

Entrar en modo config y utilizar el comando *devices device-group* para crear el grupo IOS-DEVICES, el grupo XR-DEVICE, el grupo NXOS-DEVICES y el grupo ASA-DEVICES.

Comando: *devices device-group IOS-DEVICES*

Comando: *device-name internet-rtr01*

Comando: *device-name dist-rtr01*

Comando: *device-name dist-rtr02*

Comando: *devices device-group XR-DEVICES*

Comando: *device-name core-rtr01*

Comando: *device-name core-rtr02*

Comando: *devices device-group NXOS-DEVICES*

Comando: *device-name dist-sw01*

Comando: *device-name dist-sw02*

Comando: *devices device-group ASA-DEVICES*

Comando: *device-name edge-firewall01*

Y se genera un grupo que incluya a todos los dispositivos grupo ALL.

Comando: *devices device-group ALL*

Comando: *device-group ASA-DEVICES*

Comando: *device-group IOS-DEVICES*

Comando: *device-group NXOS-DEVICES*

Comando: *device-group XR-DEVICES*

Guardar los cambios con el comando *commit* y usar el comando *end* para salir del modo de configuración, la Figura 2.36 muestra todo el proceso de agrupación de los dispositivos en los diferentes grupos.

Comando: *commit*

Comando: *end*

```
admin@ncs(config)# devices device-group IOS-DEVICES
admin@ncs(config-device-group-IOS-DEVICES)# device-name internet-rtr01
admin@ncs(config-device-group-IOS-DEVICES)# device-name dist-rtr01
admin@ncs(config-device-group-IOS-DEVICES)# device-name dist-rtr02
admin@ncs(config-device-group-IOS-DEVICES)# devices device-group XR-DEVICES
admin@ncs(config-device-group-XR-DEVICES)# device-name core-rtr01
admin@ncs(config-device-group-XR-DEVICES)# device-name core-rtr02
admin@ncs(config-device-group-XR-DEVICES)# devices device-group ASA-DEVICES
admin@ncs(config-device-group-ASA-DEVICES)# device-name edge-firewall01
admin@ncs(config-device-group-ASA-DEVICES)# devices device-group NXOS-DEVICES
admin@ncs(config-device-group-NXOS-DEVICES)# device-name dist-sw01
admin@ncs(config-device-group-NXOS-DEVICES)# device-name dist-sw02
admin@ncs(config-device-group-NXOS-DEVICES)# devices device-group ALL
admin@ncs(config-device-group-ALL)# device-group ASA-DEVICES
admin@ncs(config-device-group-ALL)# device-group IOS-DEVICES
admin@ncs(config-device-group-ALL)# device-group NXOS-DEVICES
admin@ncs(config-device-group-ALL)# device-group XR-DEVICES
admin@ncs(config-device-group-ALL)# commit
Commit complete.
admin@ncs(config-device-group-ALL)# end
admin@ncs#
```

Figura 2.35 Agrupación de los dispositivos

Es relevante recordar que al ingresar varias líneas de comandos, podría surgir confusión en relación con la ubicación actual. Para evitar este tipo de situaciones, es aconsejable recurrir al comando *pwd* con el propósito de determinar la ubicación actual.

En esta etapa, se procederá a verificar el estado de sincronización de todos los dispositivos internos, con el fin de determinar si se han producido cambios que difieran de la configuración establecida en NSO. Para llevar a cabo esta verificación, se empleará el comando *check-sync* en los grupos correspondientes. Para efectos del ejercicio práctico, esta acción se ejecutará en el grupo "IOS-DEVICES".

Comando: *devices device-group IOS-DEVICES check-sync*

La Figura 2.37 muestra los dispositivos asociados a IOS-DEVICES y como se puede observar coincide con la información ingresada en la Figura 2.36.

```
admin@ncs# devices device-group IOS-DEVICES check-sync
sync-result {
    device dist-rtr01
    result in-sync
}
sync-result {
    device dist-rtr02
    result in-sync
}
sync-result {
    device internet-rtr01
    result in-sync
}
admin@ncs#
```

Figura 2.36 Verificación de los dispositivos en un grupo

2.2.3 Explorando la red con la cli de nso

Cuando se lleva a cabo la inicialización "sync-from" en la red, NSO efectúa la recuperación integral de todas las configuraciones actuales de los dispositivos, almacenándolas en la base de datos (CDB). Esta recuperación confiere un inmenso valor agregado a NSO puesto que facilita un acceso total de las configuraciones de la red desde un único punto.

Es esencial aclarar que esta configuración no se trata simplemente de una copia de seguridad de un archivo de texto, sino que representa una configuración plenamente modelada. Cada dispositivo de red, incluyendo aquellos que son heredados y basados en interfaces de línea de comandos obtiene ahora una instantánea de configuración totalmente analizada. Esta información se extrae de los modelos NED y YANG, lo que otorga la capacidad de verificar la compatibilidad de comandos de manera precisa y eficiente.

Acceder a la CLI con el siguiente comando.

Comando: *ncs_cli -C -u admin*

Se revisa la configuración de ejecución completa para un solo dispositivo para el ejemplo práctico internet-rtr01.

Comando: *show running-config devices device internet-rtr01 config*

En la Figura 2.38 se puede observar toda la información de configuración del dispositivo.

```
admin@ncs# show running-config devices device internet-rtr01 config
devices device internet-rtr01
config
hostname  internet-rtr01
tailfned police cirmode
version   16.11
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
service call-home
login on-success log
platform console serial
no platform punt-keepalive disable-kernel-core
platform qfp utilization monitor load 80
vrf definition Mgmt-intf
address-family ipv4
  exit-address-family
!
address-family ipv6
  exit-address-family
!
!
enable password cisco
call-home
```

Figura 2.37 Información sobre la configuración del dispositivo, comunicación y conexión con NSO

Para mejorar la precisión de los datos devueltos, se puede emplear la palabra clave "de-select" como un complemento del comando, permitiendo filtrar elementos específicos, como se ejemplifica en la Figura 2.39. Al aplicar esta medida, el resultado obtenido se centra en la configuración que se utilizó para incorporar dicho dispositivo al inventario de NSO.

Comando: *show running-config devices device internet-rtr01* | *de-select config*

```
admin@ncs# show running-config devices device internet-rtr01 | de-select config
devices device internet-rtr01
 address   10.10.20.181
 ssh host-key-verification none
 authgroup labadmin
 device-type cli ned-id cisco-ios-cli-6.91
 device-type cli protocol telnet
 state admin-state unlocked
!
admin@ncs#
```

Figura 2.38 Visualización del dispositivo de elementos específicos

Si requiere obtener la lista de interfaces configuradas en el dispositivo como se muestra en la Figura 2.40 se ingresa el siguiente comando.

Comando: *show running-config devices device internet-rtr01 config interface*

```
admin@ncs# show running-config devices device internet-rtr01 config interface
devices device internet-rtr01
 config
  interface Loopback0
   description to
   no ip address
   shutdown
  exit
  interface GigabitEthernet1
   description to port1.sandbox-backend
   no switchport
   negotiation auto
   no mop enabled
   no mop sysid
   vrf forwarding Mgmt-intf
   ip address 10.10.20.181 255.255.255.0
   no shutdown
  exit
```

Figura 2.39 Vista de la configuración de la interfaz

Se puede recuperar los datos en un formato para programabilidad, como JSON o como XML, la Figura 2.41 muestra como es el código de configuración en JSON.

Comando: *show running-config devices device internet-rtr01 config interface* | *display json*

```
admin@ncs# show running-config devices device internet-rtr01 config interface | display json
{
  "data": {
"tailf-ncs:devices": {
  "device": [
{
  "name": "internet-rtr01",
  "config": {
    "tailf-ned-cisco-ios:interface": {
      "Loopback": [
    {
      "name": "0",
      "description": "to",
      "ip": {
    "no-address": {
      "address": false
    }
      },
      "shutdown": [null]
    }
      ],
      "GigabitEthernet": [
    {
      "name": "1",
      "description": "to port1.sandbox-backend",
      "negotiation": {
    "auto": true
      },
```

Figura 2.40 Vista de la interfaz en Json

2.2.4 Actualización de la configuración del dispositivo

Para este proceso de actualización se lo realizará a un solo dispositivo para poder observar los cambios. Se ingresará hasta el dispositivo para proceder con los cambios.

Comando: *ncs_cli -C -u admin*

Comando: *config*

Comando: *devices device dist-sw01*

Se debe tener presente, a partir de los módulos previos, que si surge la necesidad de efectuar una modificación, se debe emplear el comando *config.* En consecuencia, se accede a la configuración del dispositivo con el propósito de ejecutar las actualizaciones necesarias para este laboratorio.

En el transcurso del procedimiento de transición entre dispositivos, si surge la necesidad de determinar la ruta de configuración actual, se recomienda utilizar el comando *pwd.* Este comando proporciona información sobre el directorio de trabajo en el que se está operando en ese momento.

Comando: *config-device-dist-sw01*

Comando: *pwd*

Se procede a realizar como practica el cambio de configuración en la red se requiere agregar una nueva VLAN e interfaz. Ingresar las siguientes líneas de configuración.

Comando: *vlan 42*

Comando: *name TheAnswer*

Comando: *exit*

Comando: *interface Vlan 42*

Comando: *description "Answer to the Ultimate Question of Life, the Universe, and Everything"*

Comando: *ip address 10.42.42.42/24*

Comando: *exit*

Se realiza el proceso de guardado, recordar que esto se realiza con el comando *commit*.

Comando: *commit*

Se regresa al contexto superior de la información del dispositivo con el comando *top* y se verifica los cambios en el dispositivo con el comando *show configuration*, la Figura 2.42 muestra los cambios realizados en el dispositivo.

Comando: *top*

Comando: *show configuration*

```
admin@ncs(config)# show configuration
devices device dist-sw01
 config
  vlan 42
   name TheAnswer
  !
  interface Vlan42
   no shutdown
   description Answer to the Ultimate Question of Life, the Universe, and Everything
   ip address 10.42.42.42/24
  exit
 !
!
admin@ncs(config)#
```

Figura 2.41 Cambio en configuración del dispositivo

Si se tiene la intención de llevar a cabo un cambio utilizando las tecnologías API NETCONF o RESTCONF en NSO (Network Services Orchestrator). Como parte de su proceso de preparación, se puede obtener una vista previa de la versión en formato XML de la nueva configuración que se desea implementarse.

Para lograr esto, utilizaron la opción "display xml" disponible en las APIs NETCONF o RESTCONF, como se muestra en la Figura 2.43.

Comando: *show configuration* | *display xml*

```
admin@ncs(config)# show configuration | display xml
<devices xmlns="http://tail-f.com/ns/ncs">
  <device>
    <name>dist-sw01</name>
    <config>
      <vlan xmlns="http://tail-f.com/ned/cisco-nx">
        <vlan-list>
          <id>42</id>
          <name>TheAnswer</name>
        </vlan-list>
      </vlan>
      <interface xmlns="http://tail-f.com/ned/cisco-nx">
        <Vlan>
          <name>42</name>
          <description>Answer to the Ultimate Question of Life, the Universe, and Everything</description
          <ip>
            <address>
              <ipaddr>10.42.42.42/24</ipaddr>
            </address>
          </ip>
        </Vlan>
      </interface>
    </config>
  </device>
</devices>
```

Figura 2.42 Información para el uso en API

Configuraciones revertidas

En caso de que se produzca un error en la configuración y sea necesario deshacerlo, NSO ofrece una serie de pasos para revertir las configuraciones efectuadas.

NSO guarda archivos de reversión correspondientes a cada cambio realizado, lo que permite cargar una modificación específica mediante la referencia al número de identificación (ID) asociado a ese cambio. En ausencia de un ID de reversión específico, NSO recurre a la confirmación más reciente. Esta medida es especialmente útil en situaciones en las que una sola confirmación afecta a múltiples dispositivos y modifica numerosas líneas de configuración en cada uno de ellos. En tal caso, una reversión desharía todas las configuraciones realizadas en ese proceso.

Para observar las configuraciones anteriores al último cambio efectuado, se utiliza el siguiente comando *rollback*. La Figura 2.44 ilustra la información previamente almacenada en el dispositivo antes de llevar a cabo dicho cambio.

Comando: *show configuration rollback changes*

```
admin@ncs(config)# show configuration rollback changes
devices device dist-sw01
 config
  no vlan 42
  no interface Vlan42
 !
!
admin@ncs(config)#
```

Figura 2.43 Consulta del último cambio realizado para reversión

Cuando ya se tiene identificado el cambio que se realizó anteriormente se utiliza el comando *rollback configuration* para que cargue la información antes de los cambios realizados y se observa la configuración cargada mediante el uso del comando *show*.

Comando: *rollback configuration*

Comando: *show configuration*

Use el comando *commit dry-run outformat native* para examinar los comandos que se ejecutarán en el dispositivo para lograr la reversión, La Figura 2.45 muestra la información que tendrá el dispositivo antes de realizar el proceso de guardado.

Comando: *commit dry-run outformat*

```
admin@ncs(config)# commit dry-run outformat native
native {
    device {
        name dist-sw01
        data no vlan 42
             no interface Vlan42
    }
}
admin@ncs(config)#
```

Figura 2.44 Visualización de que se va a reversar correctamente el dispositivo

Una vez verificado de que el cambio es el correcto se confirma el cambio con el comando *commit*

Comando: *commit*

2.2.5 Plantillas

Hay varias formas en que NSO puede ayudar con la configuración de múltiples dispositivos, pero la forma más sencilla de comenzar es usando plantillas de dispositivos.

En este ejercicio, se creará una plantilla de dispositivo simple que se utilizará para configurar el servidor DNS.

Comience de nuevo en ncs_cli y entre en modo config.

Comando: *ncs_cli -C -u admin*

Comando: *config*

Primero, comience con una plantilla básica. Ejecute los siguientes comandos para crear una plantilla simple en NSO en la base (CDB).

Comando: *devices template SET-DNS-SERVER*

Comando: *ned-id cisco-nx-cli-5.23*

Comando: *config*

Comando: *ip name-server servers 208.67.222.222*

Comando: *ip name-server servers 208.67.220.220*

Se observa en la Figura 2.46 qué aspecto tiene esta plantilla para NSO. Vuelva con el comando *top* y emita un *show configuration* para ver los cambios pendientes.

Comando: *top*

Comando: *show configuration*

```
admin@ncs(config-config)# top
admin@ncs(config)# show configuration
devices template SET-DNS-SERVER
 ned-id cisco-nx-cli-5.23
  config
   ip name-server servers [ 208.67.222.222 208.67.220.220 ]
  !
 !
!
admin@ncs(config)#
```

Figura 2.45 Vista de la plantilla básica

NSO toma la entrada de dos servidores de nombres diferentes y los combina en un objeto de lista dentro de la base (CDB).

Guardar la nueva plantilla que se creó en la NSO.

Comando: *commit*

Ahora la plantilla generada se va a probar en un dispositivo Nexus (cisco-nx). Emita un *do show devices list* para recordar qué dispositivos son Nexus, como se muestra en la Figura

2.47. El comando *do* le permite emitir comandos de nivel superior a modo de CLI dentro de un estado o en modo de configuración.

Comando: *do show devices list*

```
admin@ncs(config)# do show devices list
NAME             ADDRESS       DESCRIPTION  NED ID                ADMIN STATE
------------------------------------------------------------------------------
core-rtr01       10.10.20.173  -            cisco-iosxr-cli-7.45  unlocked
core-rtr02       10.10.20.174  -            cisco-iosxr-cli-7.45  unlocked
dist-rtr01       10.10.20.175  -            cisco-ios-cli-6.91    unlocked
dist-rtr02       10.10.20.176  -            cisco-ios-cli-6.91    unlocked
dist-sw01        10.10.20.177  -            cisco-nx-cli-5.23     unlocked
dist-sw02        10.10.20.178  -            cisco-nx-cli-5.23     unlocked
edge-firewall01  10.10.20.171  -            cisco-asa-cli-6.6     unlocked
edge-sw01        10.10.20.172  -            cisco-ios-cli-6.91    unlocked
internet-rtr01   10.10.20.181  -            cisco-ios-cli-6.91    unlocked
admin@ncs(config)#
```

Figura 2.46 Lista de los dispositivos Nexus

Luego emita el siguiente comando *devices device dist-sw01 apply-template template-name SET-DNS-SERVER* para decirle a NSO que aplique esa plantilla a dist-sw01, como se muestra en la Figura 2.48.

Commando: *devices device dist-sw01 apply-template template-name SET-DNS-SERVER*

```
admin@ncs(config)# devices device dist-sw01 apply-template template-name SET-DNS-SERVER
apply-template-result {
    device dist-sw01
    result ok
}
admin@ncs(config)#
```

Figura 2.47 Aplicación de la plantilla al dispositivo SW-01

Ahora actualizará la plantilla para admitir dispositivos IOS y ASA también. Ingrese esta configuración en NSO:

Comando: *devices template SET-DNS-SERVER*

Comando: *ned-id cisco-ios-cli-6.91*

Comando: *config*

Comando: *ip name-server name-server-list 208.67.222.222*

Comando: *ip name-server name-server-list 208.67.220.220*

Comando: *exit*

Comando: *exit*

Comando: *device template SET-DNS-SERVER*

Comando: *ned-id cisco-asa-cli-6.6*

Comando: *config*

Comando: *dns domain-lookup mgmt*

Comando: *dns server-group DefaultDNS*

Comando: *name-server 208.67.220.220*

Comando: *name-server 208.67.222.222*

Comando: *exit*

Comando: *device template SET-DNS-SERVER*

Comando: *ned-id cisco-iosxr-cli-7.45*

Comando: *config*

Comando: *domain name-server 208.67.222.222*

Comando: *domain name-server 208.67.220.220*

Comando: *exit*

Ahora use el device-groups para enviar este cambio a toda la red, la Figura 2.49 muestra la información de los cambios realizados a cada dispositivo.

Comando: *devices device-group ALL apply-template template-name SET-DNS-SERVER*

```
admin@ncs(config)# devices device-group ALL apply-template template-name SET-DNS-SERVER
apply-template-result {
  device core-rtr01
  result ok
}
apply-template-result {
    device core-rtr02
    result ok
}
apply-template-result {
    device dist-rtr01
    result ok
}
apply-template-result {
    device dist-rtr02
    result ok
}
apply-template-result {
    device dist-sw01
    result ok
}
apply-template-result {
    device dist-sw02
    result ok
}
apply-template-result {
    device edge-firewall01
    result ok
}
apply-template-result {
    device internet-rtr01
    result ok
}
```

Figura 2.48 Cambio de configuración a todos los dispositivos

Ahora puede revisar la configuración que está lista para que NSO la envíe a los dispositivos con *commit dry-run outformat native*.

Comando: *commit dry-run outformat native*

Finalmente, envíe la configuración a los dispositivos con commit

Comando: *commit*

2.3. Práctica con el api Python de nso

La API de Python de NSO se genera dinámicamente a partir del modelo de datos de la aplicación y los NED. Es una potente API para manipular cualquier función que ofrece NSO. Dentro de esta práctica se aprenderá los conceptos básicos para leer y manipular la configuración de dispositivos con la API Python de NSO y la Comprensión del flujo típico para abrir y cerrar una transacción con la API de Python de NSO.

La administración de la automatización de red se configura actualmente con dispositivos (sin NETCONF o RESTCONF y YANG habilitados) lo que implica, en manipular datos no estructurados a través de la CLI de Cisco.

Al utilizar la API Python en NSO, permitirá realizar diversas tareas de configuración, administración y automatización en entornos de red a través de la plataforma NSO. Algunas de las funcionalidades que proporciona la API de Python de NSO incluyen:

- Configuración y gestión de dispositivos: Permite enviar comandos de configuración y gestión a dispositivos de red. Esto puede incluir la creación, modificación o eliminación de configuraciones en routers, conmutadores u otros equipos de red.
- Automatización de servicios: Permite automatizar la creación, activación y desactivación de servicios de red, lo que ahorra tiempo y minimiza errores humanos en la configuración.
- Provisionamiento de servicios: Se puede implementar rápidamente nuevos servicios en la red, como la configuración de VPNs, MPLS, entre otros.
- Obtención de información de estado: La API permite obtener información en tiempo real sobre el estado de la red y los dispositivos, lo que facilita la monitorización y el diagnóstico.
- Interacción con sistemas de terceros: La API de NSO puede integrarse con otras herramientas y sistemas de gestión.

Antes de comenzar, use los siguientes comandos para activar la instancia NSO y un dispositivo Cisco IOS del laboratorio:

Comando: *ncs-netsim --dir ~/src/netsim create-device $NCS_DIR/packages/neds/cisco-ios-cli-3.8 netsim-ios*

Comando: *ncs-setup --dest ~/src --netsim-dir ~/src/netsim*

Comando: *cd ~/src*

Comando: *ncs-netsim start*

Comando: *ncs*

Comando: *ncs_cli -C -u admin*

La NSO tardará un momento en iniciarse. Emita los siguientes comandos para que NSO aprenda la configuración de los dispositivos.

Comando: *devices sync-from*

Comando: *exit*

Una vez que se ha llevado a cabo la inicialización del dispositivo, se procede a realizar modificaciones utilizando Python dentro de NSO. Por lo general, la aplicación Python se ejecuta en la misma máquina que NSO, ya que utiliza el socket TCP IPC (comunicación entre procesadores) para establecer comunicación con el demonio de NSO.

En esta sección del laboratorio de NSO CISCO, se hará uso de la biblioteca "maagic" de NSO para representar diversos tipos de aplicaciones YANG en NSO. Esta biblioteca cuenta con métodos Python especiales que simplifican la ejecución de operaciones CRUD (Crear, Leer, Actualizar y Eliminar) en la base de datos de configuración (CDB), así como en los dispositivos presentes en NSO. La versatilidad de esta biblioteca permite una adaptación instantánea a cualquier paquete importado a NSO, ya sean paquetes NED para modelos de dispositivos o paquetes propietarios.

El proceso se iniciará con algunos ejemplos básicos. En primer lugar, se abordará el siguiente ejemplo: se verificará si el Protocolo de Descubrimiento de Cisco (CDP, por sus siglas en inglés) está habilitado en el dispositivo "netsim-iosdis".

Se genera un nuevo archivo con el siguiente nombre nso-test.py, como recomendación el archivo de este dentro de la carpeta src del NCS.

Comando: *vim src/nso-test-py*

Se copea y pega el siguiente script de Python en el archivo nso-test.py. Y guardar los cambios generados en el archivo.

```
import ncs

with ncs.maapi.single_read_trans("admin", "python", groups=["ncsadmin"]) as t:

    root = ncs.maagic.get_root(t)

    device = root.devices.device["netsim-ios"]

    cdp_result = device.config.ios__cdp.run

    print(

        "For Device {}, CDP being enabled is {}".format(device.name, cdp_result)

    )
```

Ejecute el script en la terminal.

Comando: *python3 nso-test.py*

Figura 2.49 Prueba de funcionamiento del CDP

En esta prueba se valida que el CDP está habilitado como se muestra en la Figura 2.50, un ejemplo más claro es ver las interfases levantadas en el dispositivo para ello se ingresa el siguiente script.

```
import ncs

with ncs.maapi.single_read_trans("admin", "python", groups=["ncsadmin"]) as t:

    root = ncs.maagic.get_root(t)

    device = root.devices.device["netsim-ios"]

    for interface in device.config.ios__interface:

        for if_type in device.config.ios__interface[interface]:

            if hasattr(if_type, "name"):

                print(

                    f'Device {device.name}, Interface {if_type} {if_type.name}'

                )
```

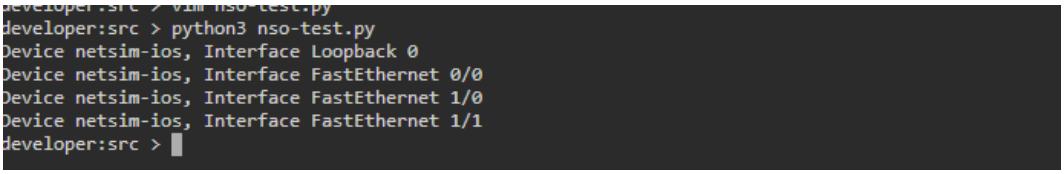

Figura 2.50 Verificación de las interfaces de netsim-ios

Como se puede observar en la Figura 2.51 se genera un listado de la consulta de las interfaces que están relacionadas con el dispositivo netsim-ios.

Los ejemplos dados eran simplemente lectura de datos, pero también es posible escribir datos.

El flujo típico de uso de la biblioteca Maagic de la API Python NSO es el siguiente:

- Crear una transacción
- Acceder a la información del dispositivo
- Manipular datos
- Aplicar configuraciones
- Cerrar la transacción

Hay varias opciones para abrir una conexión a NSO. Los más comunes de usar son los siguientes:

single_read_trans (permite solo lectura)

single_write_trans (permite lectura y escritura)

Ejecute el siguiente script de Python y analizar que realiza el script

```
import ncs

with ncs.maapi.single_read_trans('admin', 'python', groups=['ncsadmin']) as t:

    root = ncs.maagic.get_root(t)

    device_object  = root.devices.device["netsim-ios"]

    print(device_object.name)
```

with ncs.maapi.single_read_trans …, esta declaración abre una transacción de escritura en NSO (como ingresar al modo de configuración en CLI), *admin* le dice a NSO qué usuario accederá a NSO para fines de registro. El parámetro groups no es necesario en una instalación local, pero a menudo las instalaciones del sistema tienen configurado un grupo de Linux con el caso de este laboratorio *ncsadmin.*

La variable *root* se usa como un puntero o referencia, *root* se puede usar para lectura y escritura de información en la jerarquía de la CLI, por ejemplo, se puede ver el nombre del dispositivo.

device_object hace una referencia a la búsqueda del diccionario para el nombre del dispositivo de nuestro netsim en la lista de dispositivos de NSO.

root.devices.device hace referencia a todos los dispositivos en NSO CDB. Aunque el nombre dice 'dispositivo', es una lista YANG, no un dispositivo.

Si se requiere verificar los atributos de Python se utiliza print(dir())

print(dir(device_object))

3. RESULTADOS, CONCLUSIONES Y RECOMENDACIONES

1.3 Resultados

Cisco NSO (Network Services Orchestrator) es una plataforma de orquestación y automatización que permite a los ingenieros de telecomunicaciones gestionar y controlar de manera eficiente sus redes y servicios. Al utilizar NSO de Cisco, se pueden obtener una serie de beneficios y resultados. Ya que el paquete de NSO posee una interfaz gráfica la cual permite tener una visualización por medio de URL para el control de todos los dispositivos dentro de las redes la información se la puede ver en las Figuras (55-58), este utiliza para la comunicación el puerto 8080. Dentro de la interfaz gráfica, se tiene módulos que permiten realizar ciertas acciones y configuraciones a cada uno de los dispositivos.

Verificación por enlace web (parte grafica) que tiene el laboratorio y los módulos que posee para poder realizar el monitoreo como se muestra en la Figura 3.1.

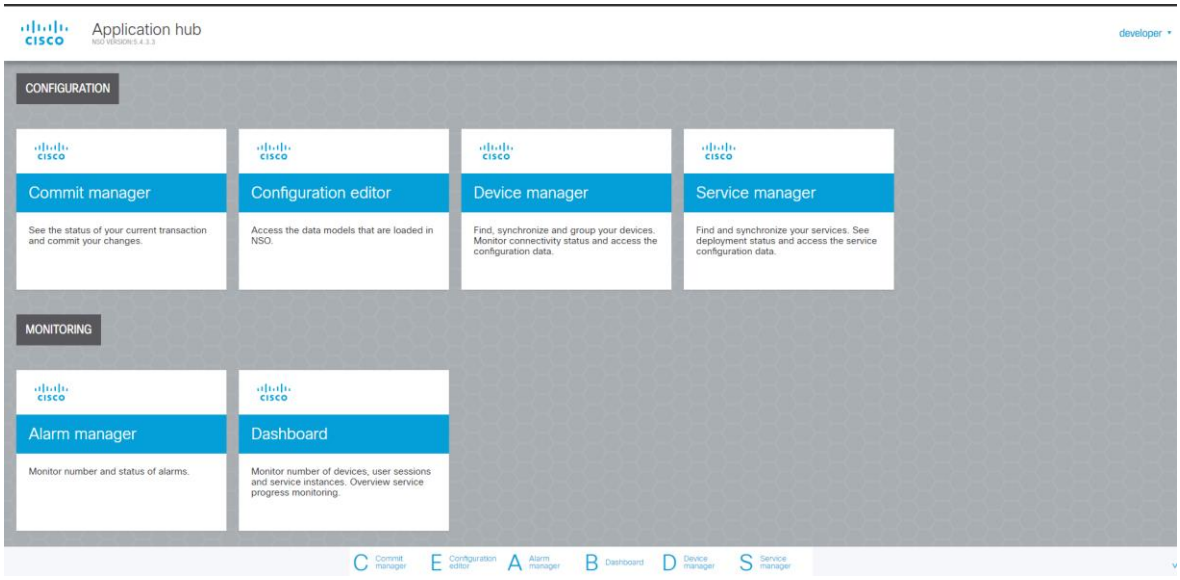

Figura 3.1 Parte Gráfica de NSO

Módulos más relevantes

Modulo Editor de Configuraciones, permite ver todas los paquetes disponibles así como los módulos que se puede usar para las configuraciones de NSO, la Figura 3.2 muestra esta información.

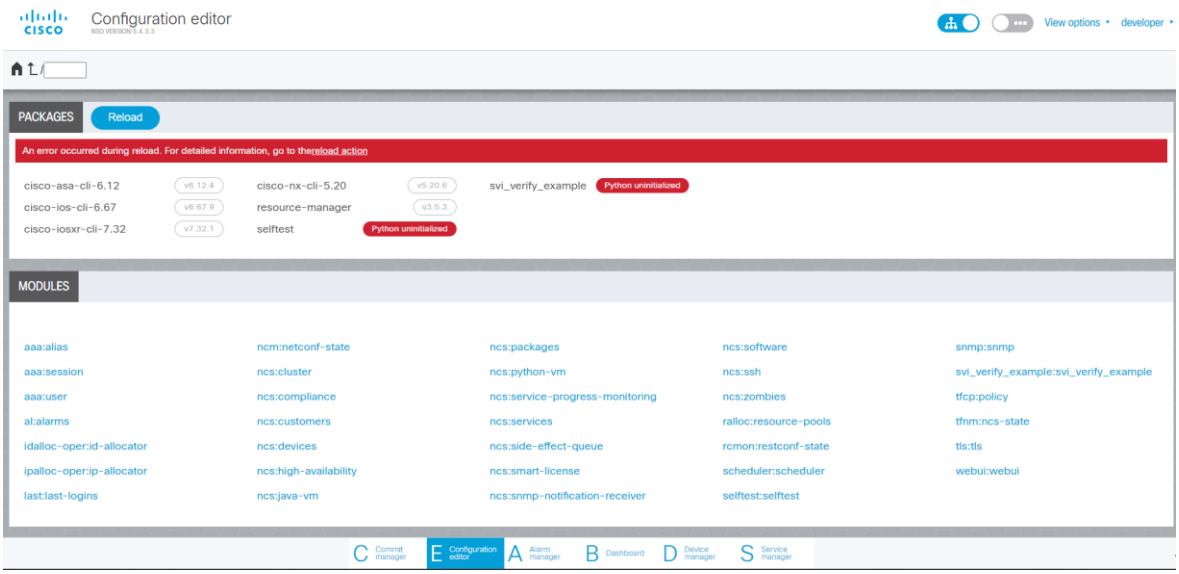

Figura 3.2 Módulo Editor de Configuraciones

Modulo Administrador de Dispositivos

Este módulo es el principal ya que aquí se puede monitorear el estado de cada dispositivo y realizar verificaciones de estado, como se muestra en la Figura 3.3.

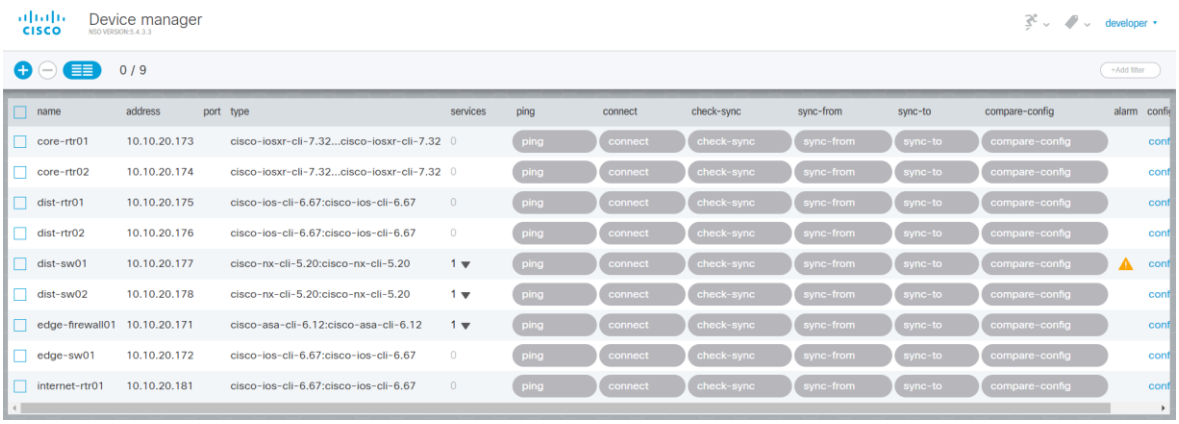

Figura 3.3 Módulo Administración de Dispositivos

Módulo Dashboard

Muestra la información a modo tablero, todo lo que posee la red en NSO; es decir los dispositivos, usuarios, alertas entre otras como se muestra en la Figura 3.4.

Figura 3.4 Dashboard NSO

Que resultados se obtiene con el manejo NSO de Cisco, considerando lo que hemos mencionado durante este laboratorio:

- Automatización eficiente: NSO simplifica y agiliza la automatización de tareas repetitivas y complejas en la configuración y gestión de la red una mejor practica para las configuraciones de las redes es utilizada la AIP Python que posee la NSO Cisco. Todo esto para reduce el tiempo necesario para implementar cambios y servicios.
- Consistencia y coherencia: NSO asegura que las configuraciones de red sean coherentes en todos los dispositivos, lo que minimiza errores humanos y garantiza una configuración uniforme en toda la red.
- Rápida implementación de servicios: NSO permite la creación rápida y sencilla de nuevos servicios, lo que acelera el tiempo de lanzamiento al mercado y mejora la capacidad de respuesta a las demandas de los clientes, al tener estructuras de configuraciones predefinidas y almacenas en los archivos de configuración.
- Flexibilidad y agilidad: La plataforma NSO permite adaptarse a cambios en los requisitos de red y servicios de manera ágil, lo que es crucial en entornos empresariales cambiantes.
- Reducción de costos: La automatización de tareas manuales y la eficiencia operativa resultante pueden reducir los costos operativos y minimizar los errores que podrían generar gastos adicionales.
- Mejora en la calidad del servicio: NSO ayuda a prevenir configuraciones incorrectas y problemas de red, lo que mejora la calidad del servicio y la satisfacción del cliente.

- Gestión centralizada: NSO permite gestionar múltiples dispositivos y servicios desde una ubicación centralizada esto mediante el manejo del Dashboard que permite a manera grafica tener una vista completa de los dispositivos, lo que simplifica la administración y el monitoreo.
- Integración con sistemas: NSO puede integrarse con sistemas de gestión y herramientas de terceros, lo que permite una mayor automatización y orquestación en el entorno TI.
- Control de versiones y auditoría: NSO mantiene un registro de los cambios realizados en la configuración, lo que facilita la auditoría y el seguimiento de versiones.

Tiene un gran dominio sobre el monitoreo de dispositivos CISCO, pero también se puede agregar varios dispositivos de otras marcas, esto gracias a que permite la integración mediante API que posee CISCO.

DIFERENCIAS

Existen otros tipos de orquestaciones de servicio por ejemplo la de Júpiter que está basado en una orquestación CSO (Servicio de orquestación de Rastros). Respecto a este laboratorio nos hemos enfocado NSO de Cisco ya que maneja un contexto más amplio sobre la orquestación y automatización de redes y servicio, mientras que la CSO de Juniper es parte de la solución Contrail (rastros) y está dirigido específicamente a la orquestación de servicios en entornos de redes definidas por software.

Ambas plataformas buscan simplificar y agilizar la gestión y la provisión de servicios en entornos de red modernos, pero están desarrolladas por diferentes proveedores y pueden tener enfoques ligeramente diferentes en función de las tecnologías.

1.4 Conclusiones

- El proceso de análisis, configuración y verificación del control de acceso en el simulador de Cisco NSO demuestra su importancia para garantizar la seguridad y la integridad de la plataforma mediante el uso de políticas detallas y reglas de acceso ya sea mediante la utilización del API Python o las configuraciones de varios dispositivos mediante una instancia de servicio. Durante este proceso, se han explorado detalladamente las políticas y reglas de acceso,

- Mediante la configuración de grupos de dispositivos, el aprovisionamiento automático de direcciones IP desde dicho grupo, y la implementación de seguridades se genera que solo los usuarios autorizados tengan acceso a las funciones y datos críticos, este enfoque de seguridad protege a la red de toparse con posibles amenazas y vulnerabilidades; estableciendo un entorno de confianza donde las operaciones de gestión y orquestación de la red pueden llevarse a cabo de manera eficiente y segura.

- Al implementar y seguir este proceso de automatización y orquestación, se fortalece la integridad de la infraestructura de red, se minimizan los riesgos de incidentes por humanos y se sientan las bases para un funcionamiento fluido de la red.

- Mediante la integración de Python y el modelando datos con el lenguaje YANG, permite a NSO generar los puntos finales de API correspondientes automáticamente, permitiendo incluir soporte para diferentes formatos de codificación y admitir otras aplicaciones, como catálogos de servicios o motores de flujo de trabajo.

- Durante este proceso, se ha evaluado detalladamente las capacidades de implementación, configuración y supervisión de diversos servicios, confirmando la versatilidad y eficacia del simulador. Al llevar a cabo estas pruebas, se ha demostrado que el simulador puede replicar situaciones del mundo real y proporcionar un entorno controlado para validar la funcionalidad de los servicios antes de implementarlos en un entorno de producción.

1.5 Recomendaciones

- Para la implementación de NSO CISCO para el monitoreo y automatización de los dispositivos de las redes, se debe tener una planificación previa detallada, una estructura de modelo de datos y los flujos de trabajo necesarios, para así poder definir los casos de uso y servicios que deseas automatizar u orquestar dentro de las redes.

- Tener un sistema de versionado de cada uno de los dispositivos que conforman la red esto para los modelos de datos y configuraciones, al poseer esta información permitirá realizar cambios controlados en los dispositivos y revertir las configuraciones en caso de ser necesario, además, permitirá comprende el modelo de datos que se usarán en NSO. Esto es crucial para definir la estructura y las relaciones de los elementos de configuración en tu red.

- Comenzar con proyectos de automatización y orquestación pequeños para tomar experiencia del manejo de NSO, como se lo ha realizado en esta práctica de laboratorio; en cada proyecto mantener siempre la documentación detallada de los modelos de datos, flujos de trabajo, scripts y procesos de automatización ya que facilitara el mantenimiento, así como también la colaboración para otros proyectos con similares configuraciones.

- Siempre implementa medidas de seguridad adecuadas para proteger los sistemas NSO, considerando siempre los siguientes factores en la seguridad de toda infraestructura de red la autenticación y la autorización, además, implementa herramientas de monitorización para supervisar la salud y el rendimiento de NSO y las redes.

4. REFERENCIAS BIBLIOGRÁFICAS

[1] «Learn NSO: The Easy Way», *Cisco DevNet Learning Labs Center*. https://developer.cisco.com/learning/labs/learn-nso-the-easy-way/setting-up-nso/ (accedido 28 de febrero de 2023).

[2] «Qué es una VPN y por qué necesitas una en 2023». https://es.vpn-mentors.com/popular/que-es-una-vpn-y-por-que-necesitas-una/ (accedido 16 de agosto de 2023).

[3] «Seguridad perimetral informática. Qué es y objetivos | Grupo Atico34». https://protecciondatos-lopd.com/empresas/seguridad-perimetral-informatica/ (accedido 17 de agosto de 2023).

[4] «¿Qué es una VPN? - Red privada virtual - Cisco». https://www.cisco.com/c/es_mx/products/security/vpn-endpoint-security-clients/what-is-vpn.html (accedido 17 de agosto de 2023).

[5] G. Flores Petlacalco, «Manual para la conexión a aplicaciones gráficas dentro del LNS», CONACYT. [En línea]. Disponible en: https://lns.buap.mx/sites/default/files/docs_sec/ManualParalaConexionConAplicacionesGraficasEneLLNS.pdf

[6] «Informática Básica: Sistemas operativos: la familia Unix», *GCFGlobal.org*. https://edu.gcfglobal.org/es/informatica-basica/sistemas-operativos-la-familia-unix/1/ (accedido 16 de agosto de 2023).

[7] «Diferencia entre linux y unix – Glosario – Significados, Conceptos, Definiciones». https://glosario.co/diferencia-entre-linux-y-unix/ (accedido 17 de agosto de 2023).

[8] E. B. School, «Lista de comandos de Linux | Euroinnova», *Euroinnova Business School*. https://www.euroinnova.ec/blog/lista-de-comandos-de-linux (accedido 16 de agosto de 2023).

[9] «3628.pdf». Accedido: 16 de agosto de 2023. [En línea]. Disponible en: https://www.tlmat.unican.es/siteadmin/submaterials/3628.pdf

[10] «Orquestación de redes: Qué es, en qué se diferencia de la gestión de redes y por qué la necesita». https://es.digi.com/blog/post/network-orchestration-vs-network-management (accedido 16 de agosto de 2023).

[11] «Descripción del YANG en dispositivos que ejecutan Junos OS | Junos OS | Juniper Networks». https://www.juniper.net/documentation/mx/es/software/junos/interfaces-telemetry/netconf/topics/concept/netconf-yang-overview.html (accedido 16 de agosto de 2023).

[12] «Configuración de NETCONF/YANG para plataformas Cisco IOS XE 16.X», *Cisco*. https://www.cisco.com/c/es_mx/support/docs/storage-

networking/management/200933-YANG-NETCONF-Configuration-Validation.html (accedido 16 de agosto de 2023).

[13] T. Cucharero Atienza, «Control y gestión de sondas de monitorización Ethernet usando NETCONF y modelos de datos YANG», masterThesis, 2017. Accedido: 16 de agosto de 2023. [En línea]. Disponible en: https://repositorio.uam.es/handle/10486/681185

[14] G. MOISIO, «YANG Data Model – Gilbert MOÏSIO», 21 de agosto de 2020. https://moisio.fr/2020/08/21/yang-data-model/ (accedido 16 de agosto de 2023).

[15] «Cisco Network Services Orchestrator (NSO)», *Cisco Developer*. https://developer.cisco.com/docs/nso/ (accedido 16 de agosto de 2023).

[16] «Introducción a XML». http://profesores.elo.utfsm.cl/~agv/elo330/2s02/projects/candia/xml.html (accedido 16 de agosto de 2023).

[17] «¿Qué es Python? - Explicación del lenguaje Python - AWS», *Amazon Web Services, Inc.* https://aws.amazon.com/es/what-is/python/ (accedido 16 de agosto de 2023).

[18] C. Gil y A. Manuel, «Automatización de redes informáticas con Python».

[19] «Network Services Orchestrator (NSO) v6.1», *Cisco Developer*. https://developer.cisco.com/docs/nso/guides/ (accedido 16 de agosto de 2023).

[20] «DevNet Sandbox - Workspace - Lab Catalog - Cisco Network Services Orchestrator». https://devnetsandbox.cisco.com/RM/Diagram/Index/43964e62-a13c-4929-bde7-a2f68ad6b27c?diagramType=Topology (accedido 16 de agosto de 2023).