UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR
DE INGENIEROS DE TELECOMUNICACIÓN

GRADO EN INGENIERIA DE TECNOLOGIAS Y SERVICIOS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

Desarrollo de escenarios virtuales para el análisis de servicios SD-WAN de seguridad en el borde de la red

IÑIGO VALENZUELA NÚÑEZ

2023

GRADO EN INGENIERÍA DE TECNOLOGÍAS Y SERVICIOS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

Título: Desarrollo de escenarios virtuales para el análisis de servicios SD-WAN de seguridad en el borde de la red

Autor: Iñigo Valenzuela Núñez

Tutor: D. Carlos Mariano Lentisco Sánchez

Departamento: Departamento de Ingeniería de Sistemas Telemáticos

MIEMBROS DEL TRIBUNAL

Presidente: D. ……

Vocal: D. ……..

Secretario: D. ……..

Suplente: D. ……..

Los miembros del tribunal arriba nombrados acuerdan otorgar la calificación de: ……

Madrid, a de de 20…

UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR DE INGENIEROS DE TELECOMUNICACIÓN

GRADO EN INGENIERIA DE TECNOLOGIAS Y SERVICIOS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

Desarrollo de escenarios virtuales para el análisis de servicios SD-WAN de seguridad en el borde de la red

IÑIGO VALENZUELA NÚÑEZ

2023

RESUMEN

La demanda de distintos servicios en la nube ha crecido de manera exponencial en los últimos años, la digitalización de distintas partes de los negocios ha llevado a los mismos a hacer más uso de aplicaciones “SaaS” e ir dejando poco a poco de lado a las aplicaciones “On Premise”. De la misma forma ha habido un gran crecimiento en la normalización del teletrabajo por parte de todo tipo de empresas. Debido a esta situación las organizaciones se han visto obligadas a reconsiderar cómo diseñan sus redes.

Los entornos de redes corporativas tienen la necesidad de proteger los tráficos dentro de la misma. Las medidas tradicionales se basan en redirigir todo el tráfico de los usuarios a través de distintas soluciones de VPNs y asegurar la red interna de la empresa con firewalls, IDS, IPS… Aunque estas soluciones sean perfectamente válidas, crean un gran cuello de botella en la red y hacen que el tráfico tenga que ser redirigido hacia la red corporativa haciendo más lento e ineficiente el proceso, a la vez los administradores de la red necesitan ofrecer soluciones flexibles que simplifiquen y centralicen el control.

Las tecnologías SASE, las cuales se integran dentro del modelo SD-WAN, presentan soluciones a estos problemas. SASE se refiere a un conjunto de tecnologías que se utilizan para proporcionar acceso seguro a los recursos de la red corporativa desde cualquier lugar. SASE se basa en la idea de que los usuarios y dispositivos móviles necesitan acceso a los recursos de la red corporativa desde cualquier lugar y en cualquier momento.

Este TFG pretende poner a prueba las distintas soluciones SASE desarrollando un escenario de ejemplo basado en SD-WAN con túneles VxLAN en el cual se harán distintas pruebas de los conceptos de SASE y de cómo se puede aliviar el consumo de recursos innecesarios en la red.

Haciendo uso del escenario desarrollado ha sido posible entender por una parte la necesidad de las redes SD-WAN junto a la facilidad del despliegue de las mismas y la importancia de la seguridad en la cadena que pasa a formar SASE. Por otra parte, se ha entendido el ahorro significativo que supone el despliegue de SASE en comparación a una red tradicional.

SUMMARY

The demand for different cloud services has grown exponentially in recent years, the digitization of different parts of the business has led them to make more use of "SaaS" applications and gradually leaving aside the "On Premise" applications. In the same way there has been a great growth in the standardization of teleworking by all types of companies. Due to this situation, organizations have been forced to reconsider how they design their networks.

Corporate network environments have the need to protect the traffic within the network. Traditional measures are based on redirecting all user traffic through different VPN solutions and securing the company's internal network with firewalls, IDS, IPS... Although these solutions are perfectly valid, they create a large bottleneck in the network and make the traffic have to be redirected to the corporate network making the process slower and inefficient, at the same time network administrators need to offer flexible solutions that simplify and centralize control.

Secured Access Service Edge (SASE) technologies, which are integrated within the SD-WAN model, present solutions to these problems. SASE refers to a set of technologies that are used to provide secure access to corporate network resources from any location. SASE is based on the idea that mobile users and devices need access to corporate network resources from anywhere and at any time.

This TFG aims to test the different SASE solutions by developing an example scenario based on SD-WAN with VxLAN tunnels in which different tests of the SASE concepts and how to alleviate the consumption of unnecessary resources in the network will be done.

By using the developed scenario, it has been possible to understand on the one hand the need for SD-WAN networks together with the ease of deployment and the importance of security in the chain that goes on to form SASE. On the other hand, it has been possible to understand the significant savings that the deployment of SASE implies in comparison to a traditional network.

PALABRAS CLAVE

- SASE
- VxLAN
- Redes Corporativas
- SD-WAN
- SDEDGE
- Teletrabajo
- NGFW
- Snort
- IDS

KEYWORDS

- SASE
- VxLAN
- Corporate Networks
- SD-WAN
- SDEDGE
- Telework
- NGFW
- Snort
- IDS

ÍNDICE DEL CONTENIDO

TABLA DE FIGURAS

1. INTRODUCCIÓN Y OBJETIVOS

1.1. INTRODUCCIÓN

Cada año el tráfico de internet experimenta crecimiento tanto en la demanda como en el número de usuarios [1]. En este contexto, la necesidad de contar con una baja latencia se vuelve imperativa, especialmente ante la aparición y generalización de prácticas como la migración del trabajo a la nube y el uso de software como servicio (SaaS) en las cuales las aplicaciones se ejecutan en servidores remotos. Según un estudio llevado a cabo en 2019, en torno al 60% de las corporaciones usan SaaS para más de la mitad de las tareas comerciales [2]. Además, debido a eventos como la pandemia del covid-19, el teletrabajo ha sufrido una aceleración exponencial a nivel mundial [3], lo que ha expuesto en muchos lugares una escasez de medios en cuanto a recursos de red [4].

Desde el año 2000 la conmutación de etiquetas multiprotocolo (MPLS) ha sido la tecnología usada como base de la infraestructura de la conexión en internet [4]. Los principales problemas que surgen debido al uso de MPLS son los costes de operación de la red y la escasa flexibilidad a la hora de modificar o desplegar nuevos puntos en la misma. Las redes WAN definidas por software (SD-WAN) surgen como una solución a los problemas de MPLS. SD-WAN cuenta con la autoconfiguración de los nodos lo que facilita y reduce el coste del despliegue de estas redes.

De media las corporaciones gastan el 10% del total de su presupuesto de tecnología en ciberseguridad [5]. La seguridad en los entornos corporativos es una pieza fundamental en la continuidad del negocio. Todos los avances que se hagan en el mundo de las redes tienen que tener en cuenta el paradigma de la ciberseguridad e idealmente promoverla.

En este TFT se propone implementar mediante la tecnología de acceso seguro a los servicios del borde de la red (SASE) una solución a los problemas planteados. En SASE existen tres elementos claves:

- Puntos de presencia, los cuales son en resumen centros de procesamiento de datos con altas capacidades de red en los que además se incluye el apartado de seguridad controlable por cada empresa cliente de este SASE.

- Los conectores de borde (sdedge) los cuales como indica su nombre se basan en servir de plataforma para acceder a los servicios SASE.

- Controlador u orquestador de SASE que permiten a los clientes ajustar y configurar SASE.

Mediante el uso de distintas tecnologías como VNX, Ryu, LXC y Wireshark se pretende crear y desarrollar un escenario, el cual permita comprobar el concepto de SASE y entender su funcionamiento. Por otra parte, entender la necesidad del cambio de las redes tradicionales a las centralizadas y definidas por software.

Para demostrar el funcionamiento de SASE se crea un escenario con un trabajador remoto que intenta acceder a recursos corporativos, aunque en principio no sería estrictamente necesario se va a emplear un túnel VxLAN para emular las posibles soluciones de VPNs que se usarían en el mundo real.

1.2. OBJETIVOS

El objetivo principal de este TFG es mediante el desarrollo de un escenario, explorar y analizar la implementación de SASE. Para lograr esto, se listan unos subobjetivos:

- Investigar y comprender en profundidad los conceptos de SASE y SD-WAN.
- Entender la necesidad de migrar a redes con control centralizado.
- Diseñar un escenario de SASE que incorpore elementos de seguridad.
- Mostrar un escenario en funcionamiento de SASE.
- Analizar los resultados y presentar conclusiones sobre SASE.

2. ESTADO DEL ARTE

2.1. REDES WAN TRADICIONALES

Las redes de amplia área, Wide Area Network (WAN), surgen de la necesidad de interconectar distintas redes locales (LAN). La red se denomina amplia porque se extiende por un área de grandes dimensiones, incluso por el mundo entero. Existen distintas tecnologías que se usan para realizar dicha conexión entre redes como líneas alquiladas, túneles o MPLS.

2.1.1. MULTIPROTOCOL LABEL SWITCHING (MPLS)

Multiprotocol Label Switching de aquí en adelante MPLS, es una tecnología para transferencia de datos a nivel IP. El estándar del protocolo de MPLS se encuentra definido en el RFC 3031 [6].

MPLS surge por dos motivaciones principales. La necesidad de una mejora en el reenvió debido al crecimiento de la demanda a nivel IP y por la misma consecuencia el tamaño de las tablas de encaminamiento IP, la segunda motivación es la necesidad de separación entre el reenvío de paquetes y las tablas de encaminamiento [7]. MPLS no fue diseñado para reemplazar IP si no para añadir reglas que permitan adaptar la misma y mejorarla.

En las redes de conmutación de paquetes, cada router toma la decisión de enrutar individualmente cada paquete. Esta decisión se fundamenta en la información contenida en la cabecera de cada paquete. El termino Forwarding Equivalency Class (FEC) se refiere a paquetes que se mandan por el mismo camino o podría entenderse que se tratan igual de cara al router, cuando un router decide enviar un paquete por un camino se engloba en esa FEC [6]. MPLS se puede usar con switches capaces de mirar y remplazar etiquetas.

MPLS permite optimizar la conmutación de paquetes haciendo uso de etiquetas para formar las distintas rutas de una red. La FEC de cada paquete se obtiene al entrar a la red ya que se marca cada ruta antes del siguiente salto [6]. Esto se traduce a eliminar la toma de decisiones según la cabecera del paquete en cada router haciendo las redes más rápidas y eficientes. Algunas de las principales ventajas de MPLS son [8] [7]:

- Ingeniería de tráfico: capacidad para determinar caminos del tráfico, rutas explicitas.
- VPN: soporte para redes privadas virtuales.
- Simplificación del núcleo de la red.
- Protección ante fallos.

La cabecera MPLS que llevan los paquetes consta de 32 bits (4 Bytes) y se encuentra situada entre el nivel 2 y el nivel 3 del modelo OSI. Los campos se dividen en:

- Etiqueta: el tamaño es de 20 bits y será el identificador principal.
- Experimental (EXP): esta etiqueta de 3 bits no está definida en el estándar con un objetivo principal, pero se suele utilizar como etiqueta para la calidad o clase de servicio.

- Último de pila (S): bit que indica si la etiqueta MPLS es la final o hay otras apiladas.
- Time to live (TTL): valor de 8 bits que define lo que el paquete debe durar antes de desaparecer.

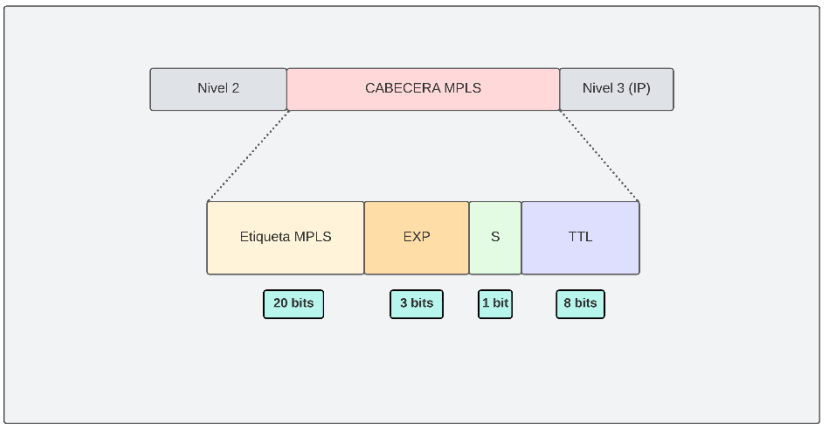


Figura 1: Descripción Cabecera MPLS

El funcionamiento a modo de resumen en MPLS reside en que los routers frontera de la red MPLS llamados "Label Switching Router" (LSR)[1] añadirán las etiquetas a los paquetes entrantes en la red MPLS. Estas etiquetas se irán conmutando a lo largo del camino seguido por el tráfico. Estos caminos, se denominan "Label Switching Path" (LSP). Un protocolo de distribución de etiquetas hará que los routers sepan que etiquetas añadir o cambiar en cada punto. Es decir, los protocolos de distribución de etiquetas permiten crear en los routers unas tablas llamadas LFIB que se usan para la conmutación de etiquetas.

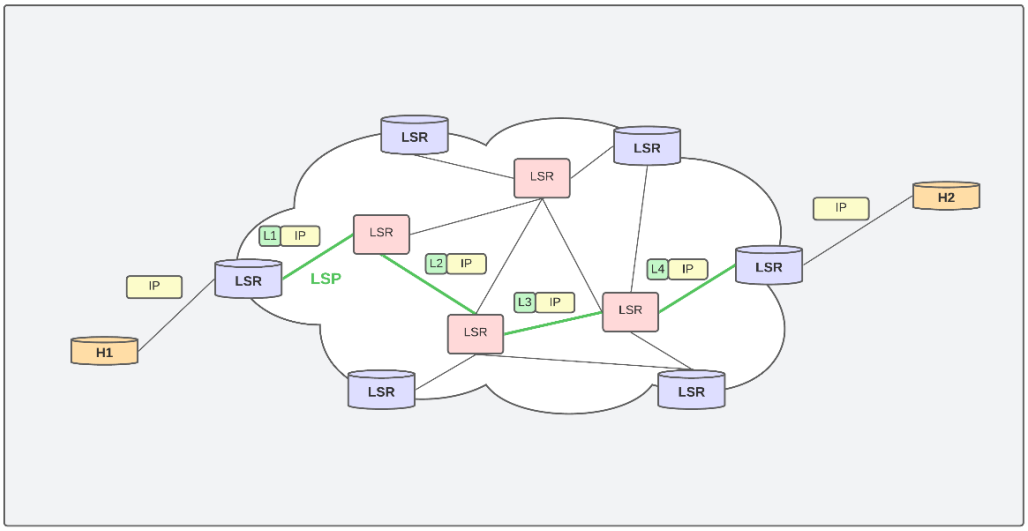


Figura 2: Ejemplo red MPLS

En la figura se muestra un ejemplo de una red MPLS, se observa que el ultimo LSR de la red MPLS entrega en paquete a H2 ya sin etiquetas. En un escenario real, el último o penúltimo salto, según la configuración de la red, quita esta etiqueta.

MPLS separa el plano de control y el de reenvío. El plano de control o encaminamiento, basado en IP, reside en la generación y actualización de tablas de forwarding. El plano de conmutación o reenvió se realiza en cada router según la LFIB. En resumen y de manera

[1] LSR se llama a cualquier router que tenga capacidad de detectar etiquetas MPLS, suelen ser routers de alta velocidad.

simplificada MPLS permite calcular la ruta inicialmente y luego ir rápidamente intercambiando las etiquetas estableciendo estos caminos etiquetados [8] [7].

2.2. SOFTWARE DEFINED WIDE AREA NETWORK (SD-WAN)

La WAN tradicional presenta distintos retos con los constantes cambios que han surgido en el mundo tecnológico, en concreto la gran demanda de acceso a internet con baja latencia y la necesidad de despliegues rápidos. Las empresas demandan una red más flexible y fácil de configurar y, además exigen sofisticados mecanismos de seguridad. Gracias a nuevos paradigmas de red como las redes SDN (Software Defined Network) y la virtualización de las funciones de red (Network Functions Virtualization) es posible transformar las redes corporativas para alcanzar esos objetivos. Destacando además la reducción de costes de montaje (CAPEX) y mantenimiento (OPEX) que derivan del uso de estas tecnologías [9].

Uno de los problemas principales de la WAN tradicional, son las redes virtuales en las que puede haber un gran número de máquinas virtuales o contenedores y la configuración correcta de las mismas se vuelve inmanejable [10]. Como resumen de las principales desventajas de la red WAN tradicional y más en concreto de MPLS son la falta de optimización para aplicaciones en la nube, difícil configuración en redes virtuales y la falta de agilidad en el despliegue [11].

En respuesta a estos problemas que surgen en la WAN tradicional, SD-WAN se presenta como una forma de optimizar tanto los recursos de la red como los costes del despliegue de la misma. En cuanto a algunas de sus principales características SD-WAN consigue una flexibilidad y escalabilidad para dotar de los recursos necesarios a distintos tipos de tráficos además de tener un rápido y sencillo despliegue de los nodos SD-WAN. A su vez la autoconfiguración de los nodos permite optimizar la red de forma que no se desaprovechen recursos disponibles y que se maximice la velocidad en la misma.

En cuanto al plano técnico de SD-WAN, se describe en primera instancia los 3 componentes principales de la arquitectura:

- Controlador: centraliza la gestión y el control de las políticas de enrutamiento y seguridad. Permite al encargado de la red observar y manejar todas las políticas desde un mismo panel. En estos controladores se determina la manera de transmitir los paquetes de datos a través de la red.
- SD-WAN EDGE: son los dispositivos que se instalan en las corporaciones o empresas que se conectan a la red SD-WAN.
- Orquestador: tareas relacionadas con la gestión y supervisión de la red. Se encarga de la revisión del tráfico y del cumplimiento de las políticas definidas en la red.

A modo de resumen final, SD-WAN hace uso del control centralizado y el aprendizaje automático para dirigir y manejar el tráfico de manera inteligente y eficiente.

2.3. SERVICE ACCESS SECURITY EDGE (SASE)

Secure Access Service Edge (SASE), se define indistintamente como una tecnología o un enfoque de arquitectura, que desempeña un papel fundamental en la seguridad y rendimiento de una red corporativa moderna. En respuesta al crecimiento exponencial de la demanda en los últimos años, SASE ofrece servicios de red en la nube y de seguridad en la nube para simplificar la conexión de los usuarios a los datos de forma fiable y segura [12]. El concepto de SASE fue utilizado por primera vez por la prestigiosa empresa dentro del mundo de la ciberseguridad Gartner, en octubre de 2019.

Es importante introducir la noción de computación de acceso múltiple en el borde (MEC o Multi-access Edge Computing). MEC se basa en acercar la computación en la nube al borde de la red [13]. Aunque el objetivo principal es reducir latencia y mejorar rendimiento estos puntos pueden ser usados para desplegar seguridad en el borde de la red.

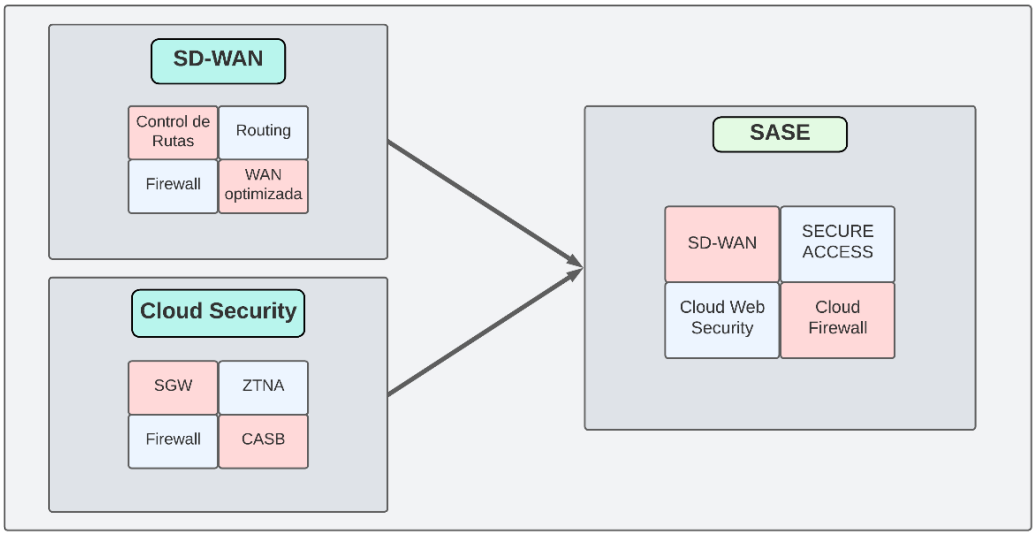


Figura 3: Mapa concepto SASE

La transformación digital está cambiando rápidamente la cultura corporativa haciendo que las redes empresariales hayan tenido que ir evolucionando con el paso de los años según las necesidades de las mismas. Hoy en día las aplicaciones y recursos usados en los entornos corporativos han migrado a la nube, aplicaciones SaaS, desapareciendo poco a poco las aplicaciones de despliegue propio ("on-premise") y por tanto necesitando una mayor seguridad, flexibilidad y escalabilidad a la hora de diseñar las redes corporativas. Por otra parte, las amenazas en el mundo de la ciberseguridad han seguido creciendo y evolucionando haciendo imperativo el uso de soluciones que aporten seguridad a la red, tanto por parte de la confidencialidad de los datos como de la disponibilidad de la misma. Al mismo tiempo que nos enfrentamos a estos retos surge la necesidad por parte de las empresas de ofrecer alternativas remotas para trabajar ya que los usuarios son cada vez más móviles y acceden a los sistemas empresariales fuera de las instalaciones. Esta explosión de necesidades ha tenido como consecuencia que las empresas tengan que volver a plantearse la arquitectura de sus

redes, enfocadas ahora a un entorno digital moderno con los requerimientos centrados en torno a la seguridad y flexibilidad.

Los principios fundamentales de SASE se basan en la convergencia de funciones de red y seguridad en una única plataforma [14]. Los Secure Access Service Edege Points Of Presence, de aquí en adelante SASE-POP, son distintos centros distribuidos que cuentan con las funciones previamente citadas de SASE. Si un dispositivo quiere acceder a estos servicios debe establecer la conexión con su SASE-POP más cercano [15]. SASE ofrece capacidades técnicas y simplifica la operabilidad de la red además de reducir el alto coste de mantenimiento y operación de las redes tradicionales.

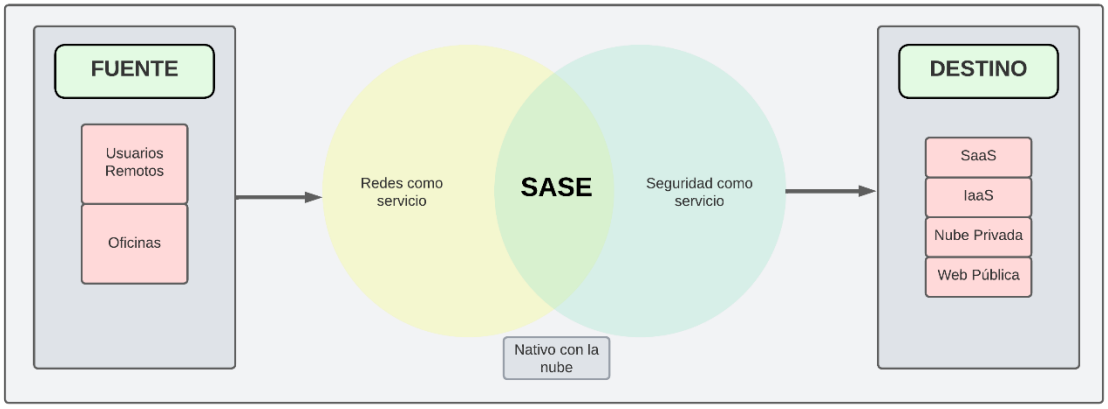


Figura 4: Descripción SASE

En este contexto se definen las funciones de SASE como ser capaz de ofrecer soluciones de red agregada y seguridad como servicio, además incluye tecnologías, muy interesantes en el ámbito de la ciberseguridad, como SD-WAN, SWG (Secure Web Gateway), CASB (Cloud Access Security Broker), NGFW (New Generation FireWall) y ZTNA (Zero Trust Network Access) más adelante se integrarán otras tecnologías como DLP (Data Loss Prevention) o ATP (Advanced Threat Protection) [12]. Como pequeña conclusión SASE es una nueva forma de plantear la arquitectura de seguridad de la red, SASE lleva la seguridad a la sesión del usuario en vez de hacer que este pase por distintos puntos de control [12].

Debido a que SASE es un concepto de arquitectura existen distintas opciones de SASE en el mercado y cada una cuenta con distintas opciones y posibilidades de configuración. Algunas de las soluciones vendidas tienen un enfoque fijado más bien en arreglar las redes tradicionales mientras que otras despliegan una arquitectura nueva en torno a las nuevas necesidades de la nube. Las soluciones implementan distintas opciones en cuanto a la seguridad, pero todas deben cumplir con esta convergencia de seguridad y optimización de red. Otras de las diferencias pueden residir en la localización y cantidad de puntos SASE-POP.

2.4. HERRAMIENTAS:

2.4.1. VIRTUAL NETWORK OVER LINUX (VNX)

VNX es una herramienta de virtualización de código abierto de propósito general diseñada para construir automáticamente entornos de prueba de redes virtuales. Permite definir y desplegar automáticamente escenarios de red compuestos por máquinas virtuales de diferentes tipos interconectadas según una topología definida por el usuario, con posibilidad de ser conectadas a redes externas. VNX ha sido desarrollado por el Departamento de Ingeniería Telemática (DIT) de la Universidad Politécnica de Madrid (UPM) [16].

VNX es una herramienta útil para probar aplicaciones o servicios de red en entornos de prueba complejos compuestos por nodos y redes virtuales, así como para crear escenarios de redes complejas que permitan a los usuarios interactuar con figuras de red realistas. VNX proporciona una forma de gestionar entornos de prueba evitando la inversión y la complejidad de gestión necesaria para crearlos utilizando equipos reales [16].

VNX consta de dos partes principales:

- Un lenguaje XML (eXtended Markup Lenguage) que permite describir el escenario de red virtual (lenguaje de especificación de VNX).
- El programa VNX, que analiza la descripción del escenario y construye y gestiona el escenario virtual en dispositivo con sistema operativo Linux.

VNX se basa en una herramienta anterior llamada VNUML (Virtual Networks over User Mode Linux), integra mejoras como plataformas de virtualización Dynamips y Olive para permitir la emulación limitada de routers CISCO y Juniper, también integra el soporte para Linux Containers (LXC) o la integración de Openvswitch con soporte para la configuración de VLAN, conexiones entre conmutadores y configuración de parámetros SDN [17].

Las principales etiquetas de VNX son:

- <global> : describe parámetros globales del escenario.
- <net> : describe las redes virtuales.
- <vm> : describe las características de cada máquina virtual2.
- <host> : describe la configuración del host del escenario principal [16].

[2] Cuando nos referimos a máquinas virtuales “vm” en la descripción del escenario se está hablando de los contenedores ligeros LXC.

```
1.
2.  <vm name="ejemplo" type="lxc" exec_mode="lxc-attach" arch="x86_64">
3.    <filesystem type="cow">rootfs_lxc_ubuntu64</filesystem>
4.    <if id="1" net="lan1">
5.      <ipv4>100.120.11.22/24</ipv4>
6.    </if>
7.    <exec seq="on_boot" type="verbatim">
8.      # Change interfaces MTU
9.      ifconfig eth1 mtu 1400
10.     sed -i -e '/iface eth1 inet static/a \   mtu 1400' /etc/network/interfaces
11.   </exec>
12.   <route type="ipv4" gw="10.20.1.1">default</route>
13.   <route type="ipv4" gw="10.20.1.1">10.20.3.0/24</route>
14.   <route type="ipv4" gw="10.20.1.1">10.20.4.0/24</route>
15.   <route type="ipv4" gw="10.20.1.1">10.30.0.0/16</route>
16. </vm>
17.
```

Figura 5: Ejemplo código VNX

2.4.2. SWITCHES SOFTWARE OPEN VSWITCH (OVS)

Open vSwitch (OVS) es un proyecto de código abierto que proporciona una solución de conmutador virtual flexible, escalable y configurable. OVS es esencial en el despliegue de algunas aplicaciones SDN. El objetivo principal de OVS es ofrecer distintas funcionalidades dentro de las redes, como soporte para VLANs, VxLANs, OpenFlow o agregación de enlaces [18].

Los switches OVS permiten, en el plano de datos, distintas opciones. En este caso se configura usando el controlador openFlow y haciendo uso de la tecnología VxLAN.

OVS basa su control y gestión en un demonio que se ejecuta en el contenedor local. Se permite por medio de comandos monitorizar y administrar el switch. Algunos de los ejemplos más relevantes son:

`ovs-ofctl show [bridge/socket]`	Información relevante de cada interfaz
`ovs-ofctl dump-tables [bridge/socket]`	Información sobre las tablas de encaminamiento

Las tablas de flujos de los conmutadores son programadas a través de mensajes que envía el controlador por su interfaz sur. El controlador realiza una programación de los switches basada en distintas aplicaciones SDN [18].

2.4.3. VIRTUAL EXTENSIBLE LAN (VXLAN)

VxLAN significa LAN extensible virtual, y define un protocolo para el establecimiento de túneles o redes superpuestas (overlay networks), utilizado para extender las capacidades de las redes LAN virtuales a través de una red de área amplia WAN, es decir conectividad de nivel 2 a través de la superposición del nivel 3 [19] [20].

La utilidad principal es permitir crear a centros de datos, empresas o corporaciones redes tradicionales de nivel 2 muchos más grandes y sin una limitación de distancia física.

VxLAN funciona encapsulando frames de nivel 2 (Ethernet) dentro de paquetes de nivel 3 usando cualquier protocolo de encaminamiento, es decir un túnel, técnicamente el túnel es sobre la capa de transporte, nivel 4 usando UDP. Esto permite extender las LANs de nivel 2 sobre otras redes de nivel 3. Las redes virtuales superpuestas creadas no necesitan ninguna configuración ya que usan la infraestructura existente en la capa subyacente. Los principales aspectos son:

- VxLAN Network Identifier (VNI): permite identificar cada VxLAN de manera similar a como funcionan los VLAN ID, pero con menor limitación de bits ya que aquí contamos con 24 bits disponibles.

- VxLAN tunnel endpoint (VTEP): es una interfaz encargada de encapsular el tráfico de nivel 2 al nivel 3. Este punto es la conexión entre la parte superpuesta y la subyacente de la red. Los distintos puntos por los que pase el tráfico y no cuenten con esta interfaz no sabrán de la existencia del túnel VxLAN, es decir será transparente para ellos. Esta puede ser implementado de manera software o hardware dentro de un router, switch o firewall que implemente soporte para VxLAN. Se puede separar en la descripción de la arquitectura las VTEP de los Network Virtualization Edges (NVE) que serían las partes encargadas de establecer el túnel y que este permanezca activo mientras se envían paquetes VxLAN [21] [19].

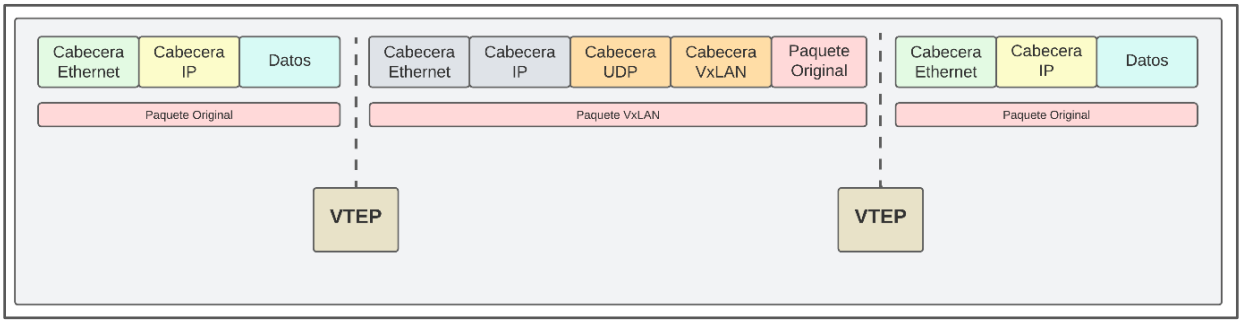


Figura 6: Ejemplo encapsulación de VxLAN en VTEP

2.4.4. Sistema de Detección de Intrusos (IDS)

Un Sistema de Detección de Intrusos, de aquí en adelante IDS, es un sistema de supervisión que está diseñado para identificar actividades sospechosas y generar alertas al detectarlas. Basándose en estas alertas se pueden tomar las medidas adecuadas para minimizar, evitar o eliminar las amenazas [22].

Los IDS son muy importantes teniendo en cuenta la gran cantidad de información confidencial y valiosa que las empresas guardan hoy en día en los sistemas informáticos y la evolución constante por parte de los atacantes en cuanto a las herramientas y exploits que utilizan. Un sistema que permita monitorizar y detectar anomalías dentro de la red es una parte esencial en cualquier red corporativa. La principal función de un IDS es alertar ante un tráfico potencialmente sospechoso.

Los IDS se basan en instalar, en un punto bien definido de la red, un sistema que permita monitorizar el tráfico y generar alertas en base a distintas reglas definidas. Los encargados de este IDS, normalmente desde el SOC (Centro de Operaciones de Seguridad), evaluaran y tomaran las medidas necesarias para las distintas alertas generadas. El problema principal de los IDS es que muchas veces generan alertas falsas y puede ser complicado ajustar los parámetros del mismo para evitarlo. Los IDS deben ser entrenados con gran cantidad de tráfico normal para poder detectar anomalías en la red [23].

El funcionamiento de los IDS depende en gran medida del tipo al que se refiera y de las capacidades del mismo. Mediante el análisis del tráfico estos IDS serán capaces de identificar patrones de ataques, estrés anormal en la red o cumplimiento de políticas. Existen dos categorías principales en los métodos de detección:

- IDS Basados en firmas: su funcionamiento consiste en la comparación del tráfico proveniente con el de secuencias conocidas como maliciosas. Este tipo de IDS permite identificar en exactitud un ataque ya que se compara directamente con otros conocidos. Aunque este método no es interesante ante ataques nuevos o con secuencias cambiadas o combinadas.

- IDS Basados en anomalías: su funcionamiento consiste en intentar detectar nuevos ataques con tan solo el estudio de la red, normalmente mediante el uso de aprendizaje automático se le enseña tráfico usual para que distinga de un posible ataque. Los falsos positivos son muy frecuentes y es frecuente la necesidad de cambios y actualizaciones en los mismos.

Normalmente se optan por sistemas híbridos que permiten ampliar las posibilidades de detectar problemas en la red [24] [23].

En cuanto a los tipos de IDS, encontramos principalmente dos categorías dependiendo de donde se encuentran los indicadores.

- Host IDS (HIDS): Detección centrada en un único dispositivo. Se analizan los logs internos y el tráfico entrante y saliente de la máquina. Soluciones comerciales como Ossec o Wazuh.

- Network IDS (NIDS): Los IDS de Red se basan en el análisis y monitorización de todos los paquetes que pasen por la red, a la que este conectada el IDS. Soluciones comerciales o de código abierto como Snort o Suricata. [25]

2.4.5. SNORT

Snort es una solución de detección de intrusos (IDS) y prevención de intrusos (IPS) de código abierto que permite el análisis y detección en tiempo real [26]. La detección de paquetes se basa en la red (NIDS). Snort es propiedad de Cisco desde 2013.

La principal característica de Snort es la facilidad de uso y la cantidad de reglas para la configuración. Existen dos grandes grupos y son las "reglas de comunidad" y las "reglas de suscripción". La diferencia principal es que las reglas de suscripción son más ágiles y probadas en cambio las de la comunidad, las cuales son gratuitas, tardan más en ser publicadas y probadas. SNORT permite buscar por código de vulnerabilidad (CVE), reglas Snort y descargarlas desde el mismo portal.

Snort permite configurar las reglas que se le dan al IDS, estas siguen una estructura que se divide en dos partes: la cabacera y las opciones de regla.

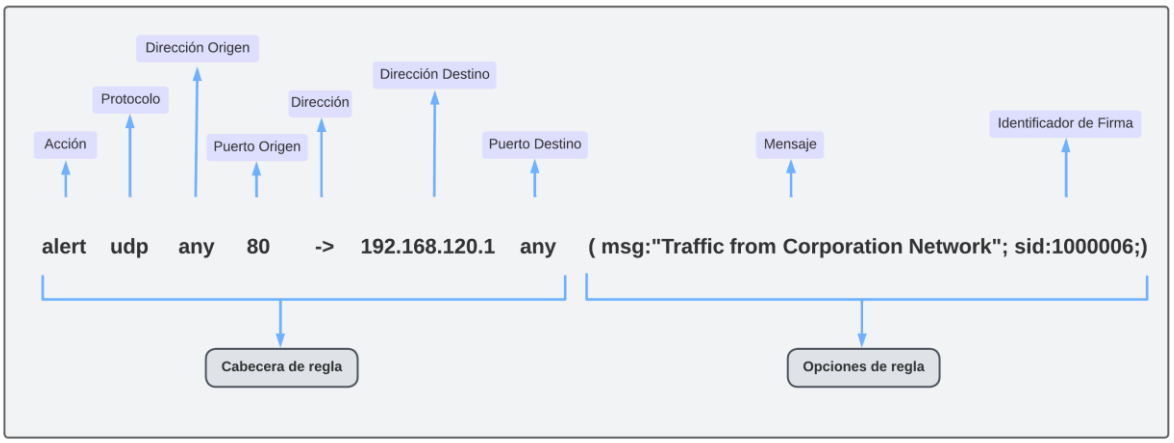


Figura 7: Campos configuración Snort

Análisis de los campos de la cabecera:

- Acción: indica que debe hacer al encontrar un paquete que cumpla con la regla. Las opciones son: alert, dynamic, pass, log, o Activate.
- Protocolo: cuales son las características de la comunicación a vigilar.
- Dirección Origen / Destino: las propiedades de una red pueden dar a se requiera vigilancia en una parte específica de esta.
- Puerto Origen / Destino: el puerto indicado en el paquete. Útil para vulnerabilidades conocidas en ciertos puertos o riesgos asociados a estos [27].

Análisis de los campos de la regla:

- Mensaje: se indica el texto que debe aparecer al saltar la alerta.
- Identificador de firma: identificador para la regla.

3. CASO DE USO

En los entornos corporativos modernos surgen nuevas necesidades de red para adaptarse a las recientes formas de teletrabajo y riesgos de ciberseguridad, en este TFG se propone el siguiente caso de uso como concepto de esta arquitectura.

Acceso de un trabajador remoto a aplicaciones corporativas haciendo uso de una arquitectura SASE. Por medio del despliegue de un SASE-POP se protegerá el acceso a los recursos, además se reducirá la latencia y al mismo tiempo que se elimina el cuello de botella dentro de la corporación dado por la arquitectura de las redes tradicionales.

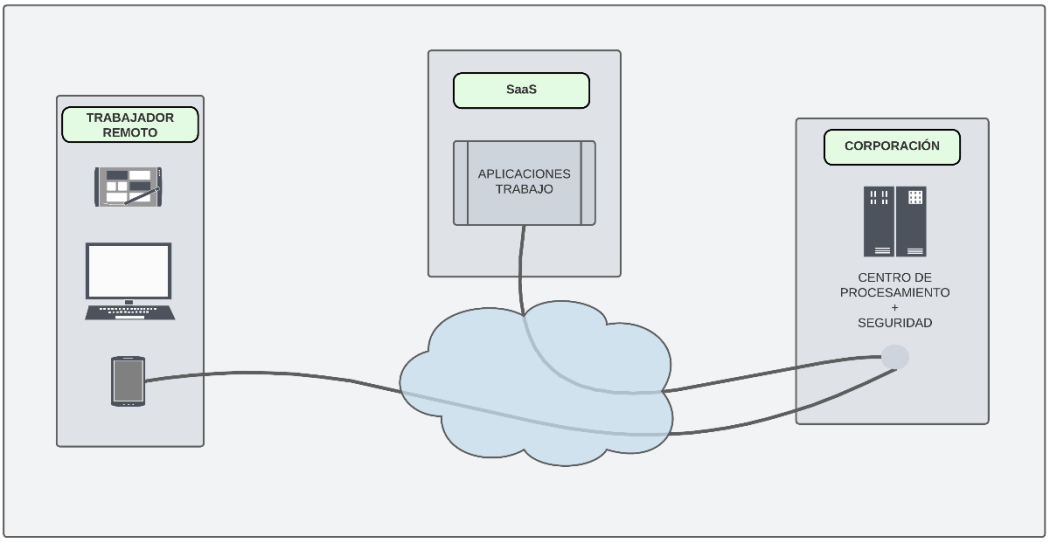


Figura 8: Ejemplo antigua red

En la figura anterior se observa la situación tradicional de las redes corporativas, de la que se pretende partir a la hora de proponer la solución del caso de uso. Como se observa todo el tráfico de los trabajadores para acceder a recursos que se encuentran fuera de la oficina, es redirigido por la red de la corporación. Lugar donde se encuentra la seguridad, provocando posibles retardos evitables y a la vez un compromiso si algún incidente ocurre en la red de la misma. A la vez las soluciones tradicionales de VPNs no permiten segmentar el acceso autorizado a los recursos.

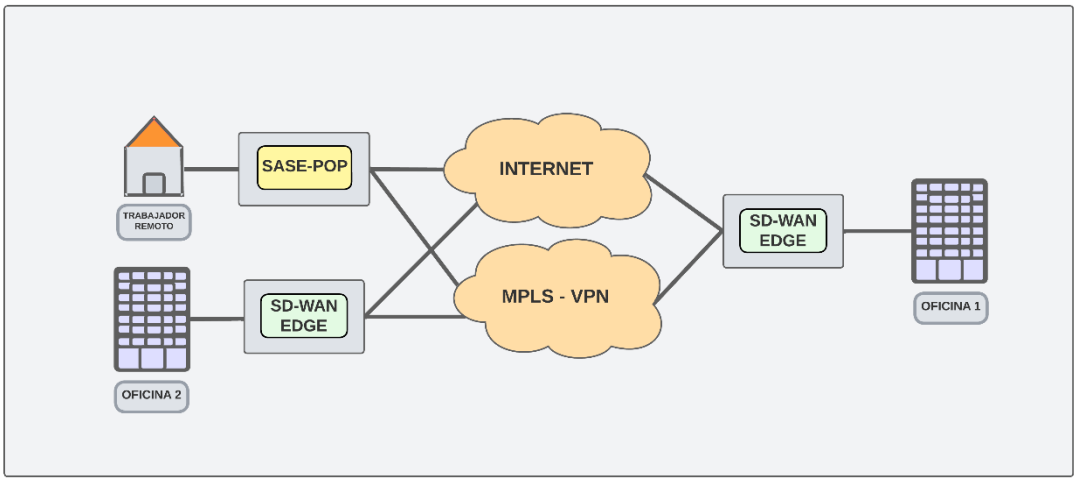


Figura 9: Concepto nueva RED

La Figura 9 pretende demostrar la nueva WAN, propuesta como solución ante los problemas de las redes tradicionales. Los puntos de acceso a la nueva red, conocidos como SASE-POPs, representan uno de los pilares esenciales de SASE en esta figura.

Distintas empresas ofrecen soluciones SASE con diversas capacidades, estas empresas crean y mantienen SASE-POPs distribuidos en distintas localizaciones por el mundo intentando, en la medida de lo posible, que las corporaciones clientes y los trabajadores remotos tengan uno de estos centros cerca. En este TFG se hace uso de esta idea para implementar un SASE-POP. En este SASE-POP se crearán los túneles VxLAN necesarios y se situarán los SDEDGE para crear la nueva arquitectura de red que permitirá instalar las reglas de seguridad necesarias. De manera habitual estos SASE-POP cuentan con distintas soluciones de seguridad y medidas las cuales los clientes podrían modificar y revisar, para este TFG se propone instalar un IDS con distintas reglas y probar el funcionamiento de la red.

La figura anterior trata de enseñar desde un punto teórico, la convergencia de los servicios de red y servicios de seguridad que se tratan como SASE. En este TFT se intenta demostrar como esta tecnología junta los bordes de la antigua red haciendo más eficiente y ágil el despliegue.

Como base del TFG se parte de esta arquitectura, en la cual se desarrolla un escenario de pruebas para implementar la seguridad y se hará uso de túneles VxLAN para conectar los distintos centros y puntos de presencia. El escenario propuesto se basa en el acceso a distintas aplicaciones por parte de un trabajador remoto y se hará especial hincapié en las dos partes esenciales de SASE, el despliegue de las redes y la seguridad. Además, se explicará de forma detallada la encapsulación del tráfico en los túneles.

4. DESARROLLO

Se decide implementar un escenario que permita observar y poner a prueba el caso de uso mencionado: Acceso de un trabajador remoto a recursos corporativos por medio de un SASE-POP.

El desarrollo de este TFG parte de un escenario de prácticas de SD-WAN desarrollado en el seno del Grupo de investigación en redes y virtualización de servicios de comunicaciones (GIROS) del Departamento de Ingeniería de Sistemas Telemáticos de la UPM. Este escenario ha sido modificado para poder tener un entorno de análisis de los servicios SASE en el marco de los servicios SD-WAN. Para ello, se ha modificado el escenario para emular un SASE-POP con funciones de IDS e IPS.

4.1. ENTORNO Y HERRAMIENTAS UTILIZADAS:

Para llevar a cabo el proyecto se decide optar por la virtualización ligera con contenedores LXC para la parte de cada dispositivo dentro de la red y por otro lado se hace uso de VirtualBox debido a que por las características del escenario y de la facilidad para la portabilidad del mismo se eligió esta opción en vez del desarrollo nativo en Linux.

- **Ryu**: es una arquitectura de trabajo basada en componentes para redes definidas por software. Ryu proporciona componentes de software con una API bien definida que facilita a los desarrolladores la creación de nuevas aplicaciones de gestión y control de redes [28].
- **VirtualBox**: solución de virtualización para distintas arquitecturas. Permite de forma sencilla virtualizar en distintos hosts con sistemas operativos como Windows, Linux o MacOS otros sistemas operativos de manera eficiente y con muchas funcionalidades de configuración [29]. VirtualBox es propiedad de Oracle.
- **Linux Containers**: herramienta de software libre que permite desde una interfaz de usuario de Linux crear e interactuar con aplicaciones en contenedores Linux. LXC está clasificado como virtualización a nivel se sistema operativo por lo que se comparte el kernel con las instancias, LXC se describe como un punto intermedio entre las máquinas virtuales tradicionales pesadas y un cambio de directorio aparente para aplicaciones como chroot [30].

4.2. DESARROLLO DEL ESCENARIO:

En esta sección se va a explicar la manera en la que se crea el escenario del TFG. Para el desarrollo de este, se parte de la Figura 9 del caso de uso. En esta figura se muestra cómo se hace uso de esta nueva WAN para acceder a las distintas aplicaciones y necesidades de red del trabajador. El escenario que se va a desarrollar pretende demostrar como este trabajador hace uso de esta red y se conecta con las distintas oficinas. En primera instancia se presenta el escenario completo que se va a desplegar y que se ira explicando de manera gradual en el desarrollo del mismo.

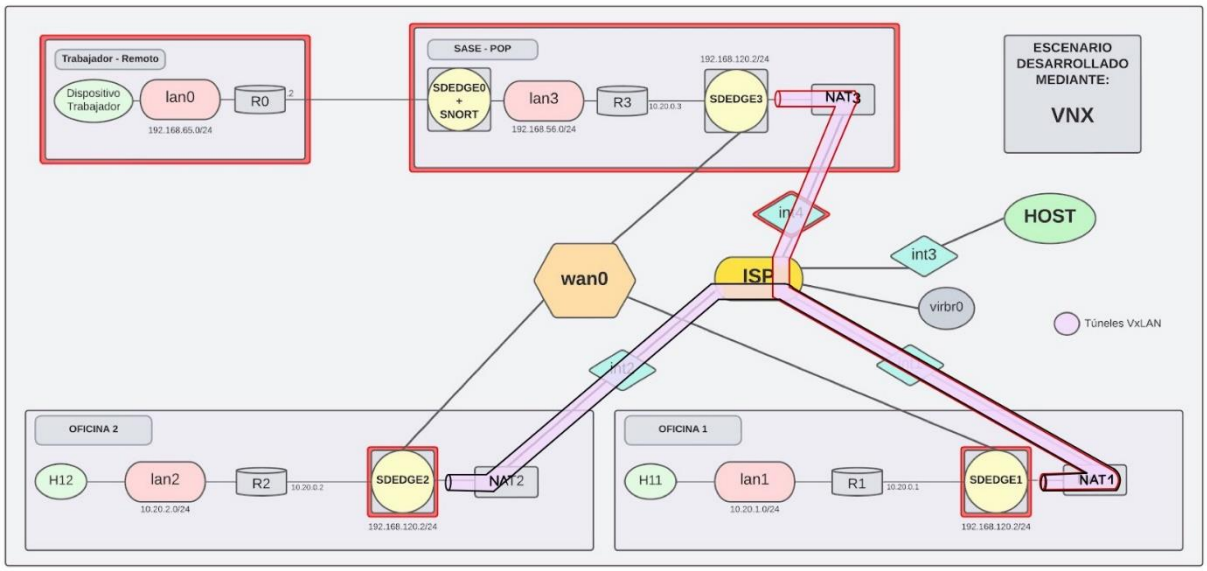


Figura 10: Escenario planteado

En la figura anterior se han marcado en rojo las principales contribuciones al escenario, aunque se han tenido que modificar algunas otras partes para el correcto funcionamiento. En esta figura, se observa cómo se añade el SASE-POP que hace de enlace con la nueva red planteada. La herramienta de VNX permite describir cada parte del escenario. Se va a explicar de manera detallada la parte del escenario añadida y una parte sobre las modificaciones anteriores. Para explicar cada parte se hará zoom sobre esta figura de forma que se pueda observar el direccionamiento.

En la parte inicial del código de VNX se indican algunas configuraciones globales como se puede observar a continuación:

4.2.1. CONFIGURACIÓN REDES:

```
0.
1.    <net name="lan1" mode="virtual_bridge" />
2.    <net name="lan3" mode="virtual_bridge" />
3.    <net name="lan0" mode="virtual_bridge" />
4.    <net name="lan31" mode="veth" type="p2p"/>
5.    <net name="lan10" mode="veth" type="p2p"/>
6.    <net name="wan3" mode="veth" type="p2p"/>
7.    <net name="lan22" mode="veth" type="p2p"/>
8.    <net name="lan32" mode="veth" type="p2p"/>
9.    <net name="int4" mode="virtual_bridge" />
10.   <net name="virbr0" mode="virtual_bridge" managed="no"/>
11.
```

Figura 11: Parte de configuraciones de redes VNX

En esta figura de código se indican algunas de las redes añadidas. A modo de explicación se indica a continuación la función de cada comando y en la siguiente figura se hace un mapa con cada una de las redes que aparecen mencionadas.

- <net … >: esta etiqueta indica la creación de una red que se basa en la unión de distintos puntos en el escenario.
- name="…": campo obligatorio. Nombre de la red, no se debe repetir el mismo ya que sirve como identificador.
- mode: campo obligatorio. Define la casuística de la red en las que encontramos dos campos posibles que son "virtual_bridge" y "veth".
    - virtual_bridge: se crea un switch virtual que conecta múltiples interfaces de red de máquinas[3].
    - veth: se resume como un túnel o tubería que conecta dos puntos. Permite que el tráfico fluya entre esas máquinas. Es una interfaz virtual.
- type: para las veth se indica que es "P2P" es decir, deben estar enganchadas a exactamente dos máquinas.

Resaltamos de este apartado dos puntos interesantes. La posibilidad de usar para la interconexión de las redes corporativas una WAN tradicional (wan0) y la conexión con el host por medio de "virbr0", generando una interfaz virtual para permitir el tráfico entre las máquinas del escenario y el host.

VNX incluye una funcionalidad muy útil para entender un escenario, que permite generar un mapa de la red completa con el comando:

```
# sudo vnx -f sase_tfg.xml -v --show-map
```

Obteniendo como respuesta el siguiente mapa:

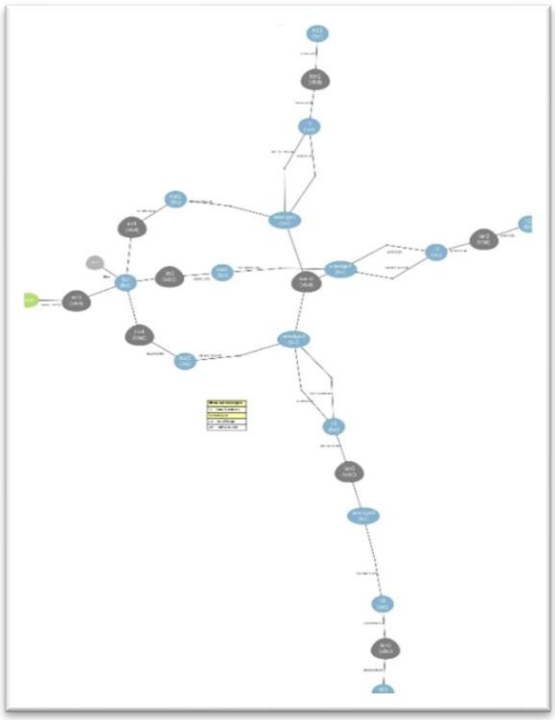


Figura 12: Mapa generado por VNX

[3] Cada vez que se mencionan "máquinas" en el trabajo, hace referencia a los dispositivos virtuales desplegados en contenedores tipo LXC en el escenario.

En este trabajo se usan direcciones de internet privadas para el tráfico dentro de las oficinas con el direccionamiento 10.20.0.0/16 haciendo uso del 10.20.1.0/24 en la red de la oficina 1 y de 10.20.2.0/24 en la segunda instalación.

4.2.2. DESARROLLO TRABAJADOR REMOTO:

En este primer desarrollo se comienza por la parte del escenario del trabajador remoto. Esta parte consta de dos dispositivos, que son el del propio trabajador y el router el cual podría simular el propio de un hogar o de un establecimiento. Estos dispositivos están conectados mediante una red de área local llamada LAN0.

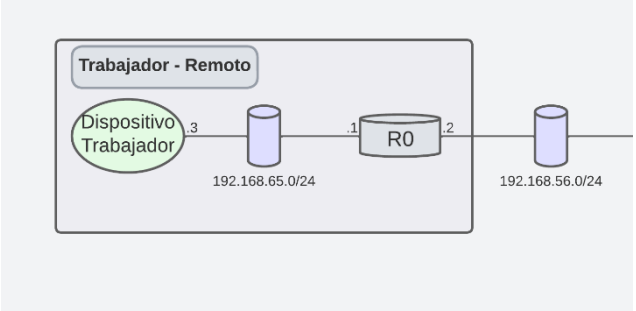


Figura 13: Trabajador Remoto

CONFIGURACIÓN DISPOSITIVO TRABAJADOR

En el siguiente punto entramos en detalle sobre el desarrollo de las máquinas que simulan a trabajadores remotos, en concreto con el LXC de h01. Este dispositivo pretende emular a un trabajador en su casa o en cualquier lugar intentando acceder a recursos mientras trabaja. El desarrollo de código de este dispositivo es similar a los otros dispositivos finales, h11 en la primera oficina y h12 en la segunda oficina.

```
0.
1.      <vm name="h01" type="lxc" exec_mode="lxc-attach" arch="x86_64">
2.        <filesystem type="cow">/rootfs_lxc_ubuntu64</filesystem>
3.
4.        <if id="1" net="lan0">
5.          <ipv4>192.168.65.3/24</ipv4>
6.        </if>
7.        <exec seq="on_boot" type="verbatim">
8.            # Change interfaces MTU
9.            ifconfig eth1 mtu 1400
10.           sed   -i   -e   '/iface   eth1   inet   static/a   \         mtu   1400'
        /etc/network/interfaces
11.       </exec>
12.       <route type="ipv4" gw="192.168.65.1">default</route>
13.     </vm>
14.
```

Figura 14: Código configuración h01

La configuración de todos los dispositivos comienza de la misma forma. Indicando el nombre de la máquina, el modo de ejecución y el archivo del contenedor a utilizar.

En la línea 4 se indica la interfaz de red a la que está conectada, en este caso solamente a la LAN0 y también se indica el rango de la misma junto a la de la interfaz especifica 192.168.65.3/24 para h01. Es decir, se indica el rango de la LAN0, 192.168.65.0/24 y la interfaz con id=1 es la .3.

El código que se encuentra entre las etiquetas <exec> al tener indicado en el parámetro “seq” “on_boot” hace que se ejecute en la maquina el código que venga dentro. Ifconfig es un comando que permite hacer ajustes en las interfaces de red de los sistemas Linux. Para las máquinas finales se configura la máxima capacidad de transmisión (MTU) a 1400 Bytes, este parámetro indica el máximo tamaño de un paquete que puede en este caso pasar por la interfaz de h01. La segunda línea del código de ejecución simplemente modifica mediante el comando “sed” un archivo de configuración de red para indicar el tamaño de la MTU. El ajuste de la máxima capacidad de los paquetes se hace para que no haya problemas al añadir el encapsulamiento con VxLAN, aunque técnicamente solo ocupan hasta 54 bytes extra en total se deja espacio suficiente para vacancia en casa necesario.

En cuanto a la configuración de las tablas de forwarding se indica la ruta que debe seguir por defecto, hacia R0 mediante la lan0.

CONFIGURACIÓN DE R0

R0 es el router ubicado en la red del trabajador, el cual está conectado por un lado a la LAN0 en la que se ubican los trabajadores remotos y por el otro a la LAN10.

```
0.
1.      <vm name="r0" type="lxc" exec_mode="lxc-attach" arch="x86_64">
2.        <filesystem
        type="cow">/usr/share/vnx/filesystems/rootfs_lxc_ubuntu64</filesystem>
3.        <if id="1" net="lan0">
4.            <ipv4>192.168.65.1/24</ipv4>
5.        </if>
6.        <if id="2" net="lan10">
7.            <ipv4>192.168.56.2/24</ipv4>
8.        </if>
9.        <exec seq="on_boot" type="verbatim">
10.           # Change interfaces MTU
11.           ifconfig eth1 mtu 1400
12.           sed   -i   -e   '/iface   eth1   inet   static/a   \         mtu   1400'
        /etc/network/interfaces
13.       </exec>
14.       <forwarding type="ip" />
15.       <route type="ipv4" gw="192.168.56.3">0.0.0.0/0</route>
16.     </vm>
17.
```

Figura 15: Código configuración R0

Este router no cuenta con acceso a la red directamente y tendrá que pasar por el SASE-POP, elemento que se explicará más adelante y en donde se ubica una NAT que traduce las direcciones.

R0 cuenta con dos interfaces de red:

- eth1 (IP: 192.168.65.1): pertenece a la LAN0 y permite la conexión con h01.
- eth2 (IP: 192.168.56.2): es la interfaz que permitirá la conexión con el SASE-POP.

Cabe destacar que de nuevo se modifica la MTU por el mismo motivo que el mencionado en la “Configuración dispositivo trabajador”. En principio no es necesario añadir esta modificación, pero de esta manera se asegura la vacancia para las cabeceras VxLAN.

4.2.3. DESARROLLO SASE-POP:

A modo de resumen la función del SASE-POP es integrar las propiedades de seguridad y de SD-WAN en un mismo punto y hacer de enlace con la nueva WAN como se indica en la Figura 9. En el caso de este escenario se ha implementado mediante cuatro contenedores LXC, SDEDGE0, SDEDGE3, R3 y NAT3, aunque el primero de ellos, SDEDGE0, con la distribución actual no es estrictamente necesario, pero se va a usar para integrar las funciones de seguridad.

La conexión entre el trabajador remoto y el SASE-POP se da mediante una LAN privada entre ambos. La idea inicial basaba el escenario en realizar un túnel extra en este paso, haciendo así un escenario más completo. En cuanto a esta LAN se puede entender como la conexión entre el router del hogar y el router que da conexión real a internet del ISP. Los SASE-POP se encuentran distribuidos por el mundo por lo que asumir que las empresas que ofrecen servicios SASE coubican sus dispositivos en las redes de los ISP es algo imaginable y que cuenta con mucho sentido.

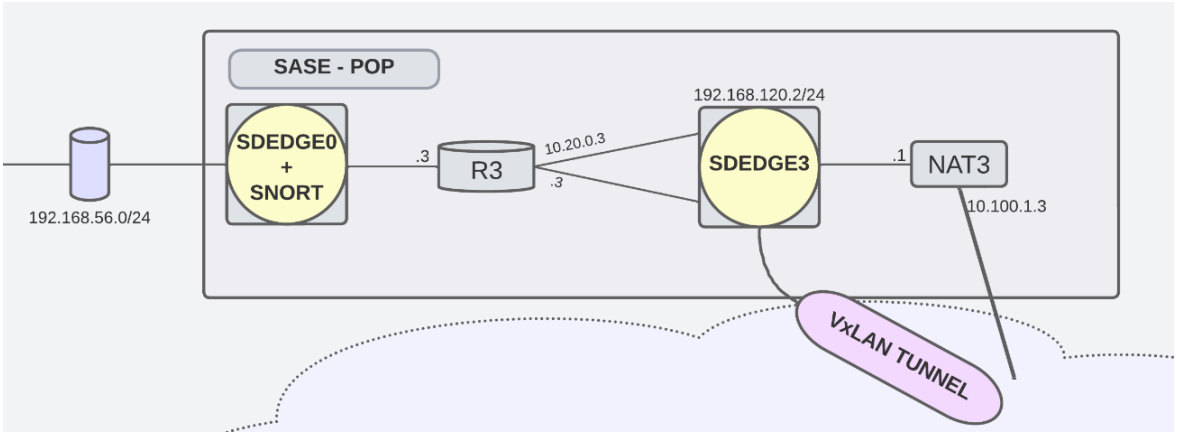


Figura 16: SASE-POP

CONFIGURACIÓN SDEDGE0-SNORT

Como primera parte del desarrollo se va a analizar el apartado de seguridad del SASE-POP. En el escenario, tras debatir sobre distintas opciones, se decide instalar una solución IDS en el SDEDGE. Este escenario pretende ser una prueba del concepto de SASE, en un entorno real se deberían instalar muchas otras soluciones partiendo por un firewall de nueva generación.

Snort es un inspector de paquetes que de manera resumida puede actuar como IDS o IPS. La decisión de instalar esta solución es por una parte la gran importancia con la que estos cuentan en el mundo de la ciberseguridad y por otra que un detector de intrusiones permite lanzar alertas según se configuren unas reglas para el tráfico lo que lo convierte en una solución elegante y suficientemente verbosa para este escenario.

La sección del SASE-POP encargada de la seguridad es el SDEDGE0. De cara a la red este SDEDGE simplemente conmuta sus interfaces por lo que no aporta una solución de red si no que se dedica a analizar el tráfico que pase por ese punto. Es importante introducir el concepto de modo promiscuo. En el mundo de las redes se habla de que una interfaz o dispositivo es promiscuo cuando puede interceptar y leer paquetes. A continuación, se introduce el código del SDEDGE0.

Existen dos principales formas de llevar a cabo la integración de Ryu con Snort. La diferencia es que en una de ellas se integran en la misma maquina y en otra se dividen en diferentes máquinas y

se conectan por un socket de red. En este TFG se ha optado por la primera opción, en esta arquitectura, Ryu recibe los paquetes de alerta de Snort a través del socket del dominio unix.

De forma resumida, la aplicación de Ryu cuenta con tres métodos nuevos. El método "__init__" que configura SnortLib para que utilice un socket de Unix e inicia el servidor. El método "packet_print" el cual imprime los paquetes que captura Snort. Por último, el método "_dump_alert" que es el que permite mostrar los mensajes diseñados en las alertas.

```
0.
1.<vm name="sdedge0" type="lxc" exec_mode="lxc-attach" arch="x86_64">
2.  <filesystem type="cow">rootfs_lxc_ubuntu64</filesystem>
3.  <if id="1" net="lan10">
4.  </if>
5.  <if id="2" net="lan3">
6.  </if>
7.  <filetree seq="on_boot" root="/etc/snort/rules">Myrules.rules</filetree>
8.  <filetree seq="on_boot" root="/etc/snort">snort.conf</filetree>
9.  <filetree seq="on_boot" root="/root/”>simple_switch_snort.py</filetree>
10.  <filetree seq="on_boot" root="/root/">simple_switch_13.py</filetree>
11.  <filetree seq="on_boot" root="/root/">ryu-50.sh</filetree>
12.  <filetree seq="on_boot" root="/root/">snort-50.sh</filetree>
13.  <filetree seq="on_boot" root="/root/">iniciaSnort.sh</filetree>
14.  <exec seq="on_boot" type="verbatim">
15.      service openvswitch-switch start
16.      sleep 5
17.      ifconfig eth1 promisc
18.      ovs-vsctl add-br br1
19.      ovs-vsctl set bridge br1 other-config:hwaddr=00:00:00:00:00:01
20.      ovs-vsctl set bridge br1 other-config:datapath-id=0000000000000001
21.      ovs-vsctl set-controller br1 tcp:127.0.0.1
22.      ovs-vsctl add-port br1 eth1
23.      ovs-vsctl add-port br1 eth2
24.      ip link set br1 up
25.
26.  </exec>
27.</vm>
28.
```

Figura 17: Código configuración SDEDGE0

El código de SDEDGE0 permite observar muchas diferencias con respecto al código de los otros SDEDGE. Empezando por la falta de interfaces y el no tener ningún túnel VxLAN. El SDEDGE0 se encuentra entre "LAN10" y "LAN3" ilustramos la situación del mismo en la siguiente figura.

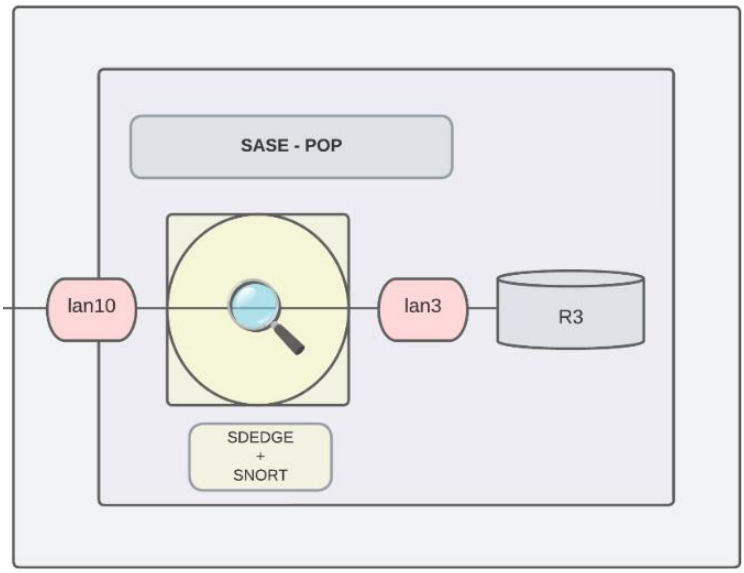


Figura 18: Función de SDEDGE0

La figura anterior representa como el SDEDGE0 es realmente una conexión directa entre lan10 y lan3. La parte relevante de este punto es la de inspeccionar los paquetes dentro del mismo.

En SDEDGE0 solo contamos con un bridge a diferencia de en los otros. En este bridge se añaden los dos puertos que tenemos y luego se levanta el mismo. La configuración de SDEDGE0 es más sencilla que las otras. Se observa cómo se configura la interfaz eth1 en modo promiscuo. Un punto interesante es que, pese a que se añade un controlador en el bridge 1, este no es estrictamente necesario para usar Snort.

En el código se incluyen siete archivos, pero solo se utilizan los cinco relacionados con Snort. Los archivos simple_switch13.py y ryu50.sh sirven para otra configuración previa a la actual. Se explica de forma detallada el contenido de los mismos.

- **Myrules.rules**

```
0.
1.# Regla para ping ICMP
2.alert icmp any any -> any any (msg:"Haciendo ping...!"; sid:1000004;)
3.
4.# Regla para el puerto TCP 80
5.alert tcp any 80 -> any any (msg:"Acceso al puerto 80 detectado!";
   sid:1000003;)
6.
7.# Regla para tráfico desde la Red de la Corporación
8.alert ip any any -> 10.20.1.0/24 any (msg:"Trafico desde la Red de la
   Corporacion!"; sid:1000005;)
9.
10.# Regla para conexión SCP (generar una alerta)
11.alert tcp any 22 -> any any (msg:"Se detecto una conexion SCP!";
   sid:1000002;)
12.
13.# Regla para transferencia de archivos SCP (registrar el archivo)
14.log tcp any 22 -> any any (msg:"Transferencia de archivo SCP!";
   content:"SCP"; nocase; file_data; sid:1000008;)
15.
```

Figura 19: Código configuración Myrules.rules

Este primer archivo es el que cuenta con reglas propias que pueden ser modificadas y personalizadas por cada usuario de Snort.

En el caso del escenario se han añadido cuatro reglas principales:

1. La primera regla es simple y se basa en que avise cada vez que pasa tráfico ICMP de cualquier tipo y a cualquier destino.
2. La segunda regla mira si el puerto destino es el 80, esta regla podría añadirse para vulnerabilidades conocidas en puertos concretos.
3. Regla para el tráfico hacia la red de la oficina 1. Comprobando el rango de direcciones IP.
4. Por último, contamos con una regla para hacer un ejemplo del uso de otros protocolos en este caso SCP. La sintaxis de estas reglas se encuentra explicada en la Figura 7: Campos configuración Snort.

Es importante comprender que estas reglas son una prueba de concepto para este TFG y que para una solución real se usan normalmente alertas depuradas entorno a vulnerabilidades conocidas o problemas conocidos del sistema.

- **snort.conf**

```
0.
1.# Archivo de configuración de Snort
2.
3.# Configurar la interfaz de red en la que Snort debe escuchar
4.var HOME_NET any
5.
6.# Configurar la red o rango de direcciones IP de la red externa
7.var EXTERNAL_NET any
8.
9.# Configurar otros preprocesadores según sea necesario
10.preprocessor stream5_global: track_tcp yes, track_udp yes
11.
12.# Configurar reglas de detección
13.include /etc/snort/rules/Myrules.rules
14.
15.# Configurar opciones de registro (logging)
16.var RULE_PATH /etc/snort/rules
17.var SO_RULE_PATH /etc/snort/so_rules
18.var PREPROC_RULE_PATH /etc/snort/preproc_rules
19.
20.# Especificar el directorio y los nombres de archivo de registro
21.var LOGDIR /var/log/snort
22.var LOGFILE snort.log
23.# Habilitar la salida de registro de paquetes
24.output unified2: filename snort.u2, limit 128
25.
```

Figura 20: Código configuración snort.conf

El archivo snort.conf es la base para el funcionamiento del mismo. En este archivo se han incluido las configuraciones globales como la red, el rango de direcciones o el tipo de preprocesadores que se usan. Estos ayudan a Snort a interpretar los paquetes de la red, permitiendo detectar y alertar sobre posibles amenazas o anomalías.

En snort.conf se deben añadir los archivos de reglas que se van a usar. En este caso en la línea 13 se añade el archivo Myrules.rules descrito previamente. Al instalar Snort este incluye miles de reglas creadas por la comunidad sobre vulnerabilidades recientes. Estas reglas pueden ser añadidas en este paso de la misma forma que se hace con el archivo de Myrules.rules.

- **simple_switch_snort.py**

El último archivo de Snort es una aplicación de ryu. Este archivo cuenta con un simple_switch por un lado y también incorpora librerías de Snort gracias a las cuales se puede imprimir las alertas configuradas en los archivos descritos.

Debido a un problema con las alertas, relacionado con el formato del texto y la conversión del mismo se tuvo que modificar ligeramente la parte de imprimir las alertas para que solo se imprima la misma hasta llegar al carácter “!”.

```
0.
1.    @set_ev_cls(snortlib.EventAlert, MAIN_DISPATCHER)
2.    def _dump_alert(self, ev):
3.        msg = ev.msg
4.        alertmsg = ''.join([str(a) for a in msg.alertmsg])
5.        if '!' in alertmsg:
6.            alertmsg = alertmsg.split('!')[0] + '!'
7.        print('alertmsg: %s' % alertmsg)
8.        self.packet_print(msg.pkt)
9.
```

Figura 21: Modificación código simple_switch_snort.py

- **snort-50.sh**

Este archivo simplemente contiene el código para iniciar el simple_switch_snort.py para comodidad a la hora de iniciar el escenario. Código para inicia el Ryu.

```
0.
1.#!/bin/bash
2.ryu-manager    /root/flowmanager/flowmanager.py    ~/simple_switch_snort.py
   ryu.app.ofctl_rest
3.
```

Figura 22: Código snort-50.sh

- **iniciaSnort.sh**

Es un archivo similar al anterior el cual inicia Snort. Para iniciar Snort se debe acceder a la consola con permisos de administrados usando por ejemplo <<sudo -i>> antes de ejecutar este Shell script.

```
0.
1.#!/bin/bash
2.snort -i eth1 -A unsock -l /tmp -c /etc/snort/snort.conf
3.
```

Figura 23: Código iniciaSnort.sh

Para iniciar de modo adecuado el escenario se deben abrir dos consolas del SDEDGE0. Primero se debe empezar ejecutando el Snort y a continuación una vez haya arrancado, pasar a iniciar Ryu[4].

CONFIGURACIÓN R3

R3 es el router situado en el SASE-POP. Este router en un escenario real debería de encargarse de las conexiones con los dispositivos externos a la red y enlazarlos con el SDEDGE correspondiente, en este caso SDEDGE3. El router R3 se define en el fichero XML de VNX tal y como muestra la Figura 24. En esta se observa como R3 cuenta con tres interfaces, una de ellas para la conexión con SDEDGE0 y las otras dos para el SDEDGE1.

[4] Una comprobación recomendada es cerciorarse de que eth1 está en modo promiscuo. Esto se puede ver con *ifconfig* en los parámetros de la interfaz debe aparecer eth1 como “PROMISC”.

```
0.
1. <vm name="r3" type="lxc" exec_mode="lxc-attach" arch="x86_64">
2.     <filesystem type="cow">rootfs_lxc_ubuntu64</filesystem>
3.     <if id="1" net="lan3">
4.         <ipv4>192.168.56.3/24</ipv4>
5.     </if>
6.     <if id="2" net="lan31">
7.         <ipv4>10.20.0.3/24</ipv4>
8.     </if>
9.     <if id="3" net="lan32">
10.         <ipv4>192.168.120.3/24</ipv4>
11.     </if>
12.     <forwarding type="ip" />
13.     <route type="ipv4" gw="10.20.0.1">10.20.1.0/24</route>
14.     <route type="ipv4" gw="192.168.56.2">192.168.65.0/24</route>
15.     <route type="ipv4" gw="192.168.120.1">0.0.0.0/0</route>
16. </vm>
17.
```

Figura 24: Extracto configuración de R3

En las rutas definidas, la parte de interés reside en la conexión con el SDEDGE3. Se observa como para acceder a este se tienen dos interfaces. La LAN31 se usa para el acceso a la oficina y para el resto se usa la LAN32. Las interfaces de R3 son:

- eth1 (IP: 192.168.56.3): pertenece a la LAN3 y permite la conexión con R0.
- eth2 (IP: 10.20.0.3): es la interfaz que permitirá la conexión entre las distintas sedes. Conectada a SDEGE3.
- eth3 (IP: 192.168.120.3): interfaz para tráfico de acceso a internet, conexión con el host y ruta por defecto. Conectada a SDEDGE3.

Se hace la comprobación de la misma utilizando la herramienta de tcpdump en la interfaz 3 de R3 y haciendo “ping” desde h01 a distintos puntos de la red.

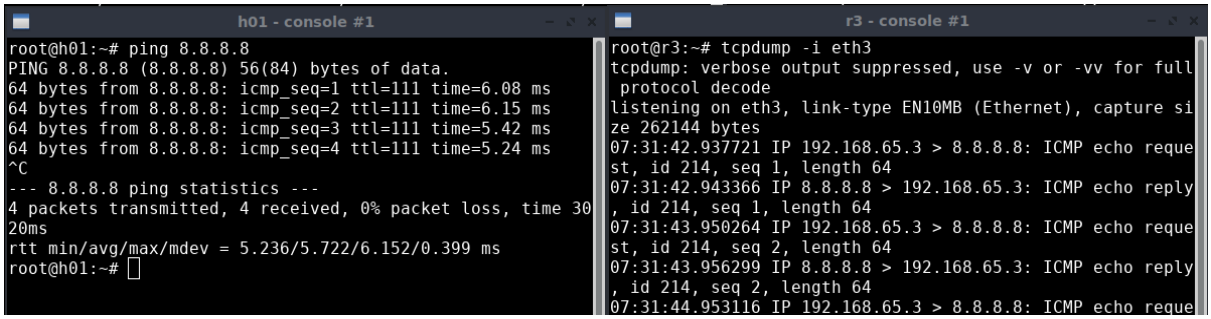


Figura 25: Comprobación interfaces R3-1

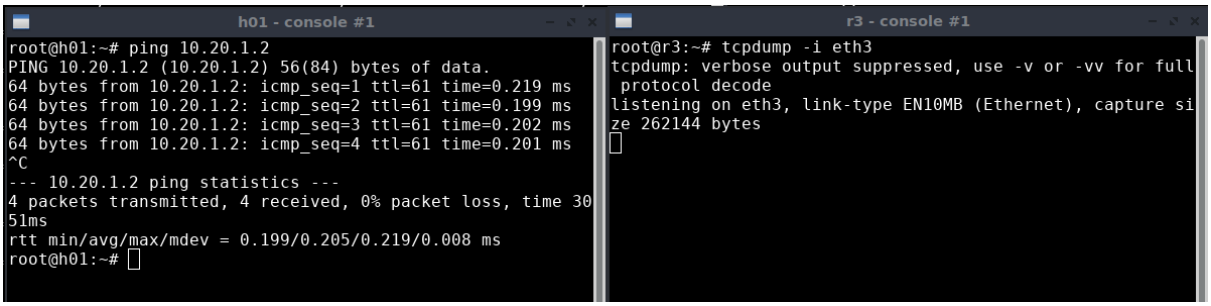


Figura 26: Comprobación interfaces R3-2

En estas imágenes se observa mediante el comando “tcpdump -i eth3” como efectivamente se usa una interfaz u otra dependiendo del destino, en este caso eth3 sería la interfaz que en un hogar cotidiano conecta con la NAT y da acceso a internet y eth2 es que se usa en la interconexión de ambas oficinas y los trabajadores remotos. Es importante resaltar que la wan0 existe en este contexto como la red tradicional que se piensa sustituir por una más flexible, aunque debido a la posibilidad de control que ofrece SD-WAN se podría forzar a encaminar los flujos por ella.

Por último, aunque no se muestra en el extracto de código, se vuelve a configurar el máximo tamaño de los paquetes que se transmiten a 1400 Bytes para que no se sobrepase el tamaño máximo al encapsular con VxLAN.

CONFIGURACIÓN SDEDGE3

La configuración más complicada se encuentra en los sdedge de la red. Estos dispositivos van a ser la clave en la encapsulación y la interconexión entre las sedes y los trabajadores remotos.

Cada SD-WAN Edge se presenta como un router, es decir de capa 3. Sin embargo, también incorpora un switch Open vSwitch que opera como un conmutador SDN OpenFlow. Este conmutador está controlado por una instancia del controlador SDN Ryu que se ejecuta localmente en el mismo contenedor.

```
0.
1. <vm name="sdedge3" type="lxc" exec_mode="lxc-attach" arch="x86_64">
2.     <filesystem type="cow">rootfs_lxc_ubuntu64</filesystem>
3.     <if id="1" net="wan0">
4.     </if>
5.     <if id="2" net="wan3">
6.     </if>
7.     <if id="3" net="lan31">
8.     </if>
9.     <if id="4" net="lan32">
10.     </if>
11.     <filetree seq="on_boot" root="/root/">simple_switch_13.py</filetree>
12.     <filetree seq="on_boot" root="/root/">ryu-50.sh</filetree>
13.     <exec seq="on_boot" type="verbatim">
14.         service openvswitch-switch start
15.         sleep 5
16.         ovs-vsctl add-br br1
17.         ovs-vsctl add-port br1 eth2
18.         ovs-vsctl add-port br1 eth4
19.         ip addr add 192.168.120.2/24 dev br1
20.         ip link set br1 up
21.         ip route add 0.0.0.0/0 via 192.168.120.1
22.         ovs-vsctl add-br br0
23.         ovs-vsctl set-fail-mode br0 secure
24.         ovs-vsctl set bridge br0 other-config:hwaddr=00:00:00:00:00:01
25.         ovs-vsctl set bridge br0 other-config:datapath-id=0000000000000001
26.         ovs-vsctl set-controller br0 tcp:127.0.0.1
27.         ip link add veth0 type veth peer name veth1
28.         ifconfig veth0 up
29.         ifconfig veth1 up
30.         ip link add vxlan2 type vxlan id 1 remote 10.100.1.1 dstport 4789
31.         ip link set vxlan2 up
32.         ovs-vsctl add-port br0 vxlan2
33.         ovs-vsctl add-port br0 eth1
34.         ovs-vsctl add-port br0 eth3
35.         ovs-vsctl add-port br0 veth0
36.         ovs-vsctl add-port br1 veth1
37.     </exec>
38.  </vm>
39.
```

Figura 27: Código configuración SDEDGE3

En la primera instancia se hace la configuración de las interfaces, de forma análoga a las otras partes de la red. SDEDGE3 se encuentra situado entre R3 y NAT3, aunque a continuación se explicara con más detalle. La configuración inicial de las interfaces es la siguiente:

- eth1: interfaz que pertenece a la WAN0.
- eth2: interfaz de la WAN3.
- eth3: interfaz de la LAN31.
- eth4: interfaz de la LAN32

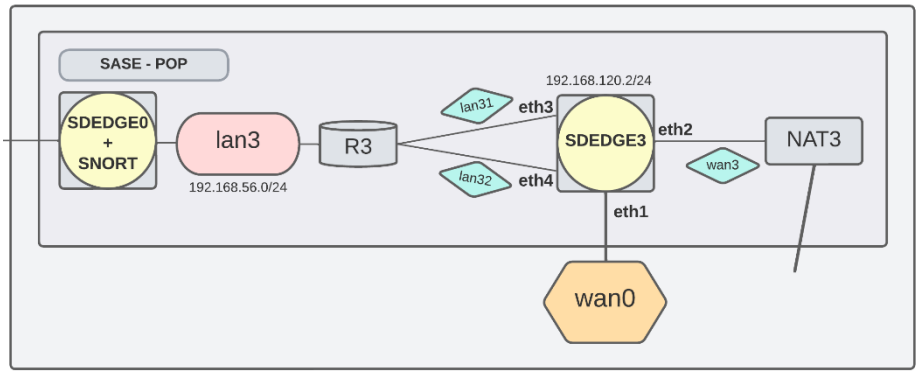


Figura 28: Mapa configuración SDEDGE3

Como se puede observar la Figura 10, la parte a la derecha de R3 es parecida la de las oficinas 1 y 2, usando el SDEDGE3 como punto donde se instala el controlador y ejecutamos la aplicación de Ryu de un simple_switch.

Entramos primero a la explicación del código que se encuentra dentro de la etiqueta <exec> el cual como se ha comentado antes se ejecuta al iniciar el contenedor.

- Línea 14: *service openvswitch-switch start*

  Se inicia el servicio de openVswitch, un conmutador. Se puede comprobar el funcionamiento del mismo con comandos como <<ovs-ofctl show "interfaz">>.

- Línea 15: *sleep 5*

  La ejecución del código se para 5 segundos para esperar al conmutador openVswitch.

- Línea 16: *ovs-vsctl add-br br1*

  Se crea un nuevo puerto, br1, en SDEDGE. Estos bridges de OVS permiten conectar internamente distintos puertos tradicionales con puertos virtuales.

- Línea 17: *ovs-vsctl add-port br1 eth2*
- Línea 18: *ovs-vsctl add-port br1 eth4*

  Se añaden los puertos eth2 y eth4 al bridge creado en el paso anterior, br1.

- Línea 19: *ip addr add 192.168.120.2/24 dev br1*
- Línea 20: *ip link set br1 up*

  Se hace una configuración final de br1. Se añade la dirección IP 192.168.120.2 y se activa el bridge 1, br1.

- Línea 22: *ovs-vsctl add-br br0*
- Línea 23: *ovs-vsctl set-fail-mode br0 secure*
- Línea 24: *ovs-vsctl set bridge br0 other-config:hwaddr=00:00:00:00:00:01*
- Línea 25: *ovs-vsctl set bridge br0 other-config:datapath-id=0000000000000001*

Se crea y configura un segundo ovs bridge en el SDEDGE1, br0. Otras de las configuraciones son activar el modo de seguridad ante fallos y añadir manualmente una dirección MAC y un datapath-id.

- Línea 26: *ovs-vsctl set-controller br0 tcp:127.0.0.1*

Se conecta el controlador al bridge 0 (br0), para añadir se necesita una dirección IP. En este caso se usa la 127.0.0.1. Para comprobar que el controlador se añade correctamente usamos el comando: <<ovs-vctl list controller>>

```
root@sdedge3:~# ovs-vsctl list controller
_uuid               : eb7606b7-9d6f-44ff-af6c-ed6357adb9a9
connection_mode     : []
controller_burst_limit: []
controller_queue_size: []
controller_rate_limit: []
enable_async_messages: []
external_ids        : {}
inactivity_probe    : []
is_connected        : true
local_gateway       : []
local_ip            : []
local_netmask       : []
max_backoff         : []
other_config        : {}
role                : other
status              : {last_error="Connection refused", sec_since_connect="7900"
, sec_since_disconnect="8021", state=ACTIVE}
target              : "tcp:127.0.0.1"
type                : []
root@sdedge3:~#
```

Figura 29: Comprobación controlador

- Línea 27: *ip link add veth0 type veth peer name veth1*
- Línea 28: *ifconfig veth0 up*
- Línea 29: *ifconfig veth1 up*

Se crean dos interfaces virtuales, veth0 y veth1, las cuales están conectadas entre sí. En las siguientes líneas se activan con el comando “up”.

- Línea 30: *ip link add vxlan2 type vxlan id 1 remote 10.100.1.1 dstport 4789*
- Línea 31: *ip link set vxlan2 up*

En este punto es en el que se crea la interfaz para el encapsulamiento VxLAN. Debido a que desde e1 SASE-POP sale un túnel hacia la oficina 1 solo se necesita una interfaz VxLAN.

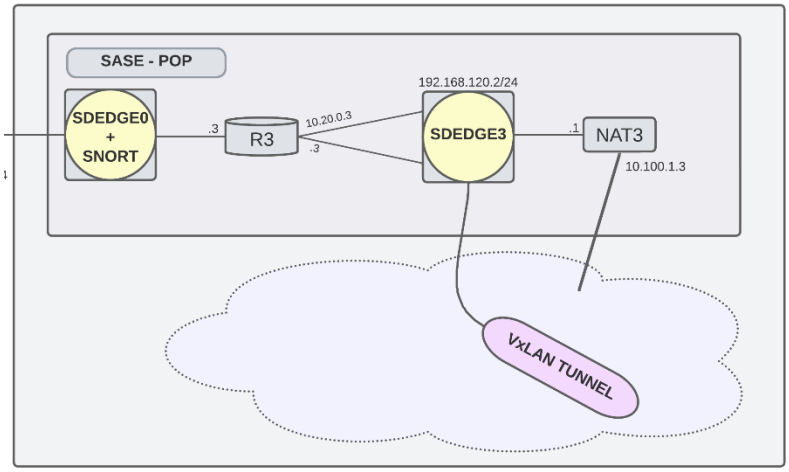


Figura 30: Túnel desde SDEDGE3

Los comandos para la creación de las VxLANs han tenido que ser modificados para poder añadir el identificador y no con la opción general que se basa en claves, con el argumento de <<options:Key=40>>. Por otra parte, se añade el puerto destino el cual para VxLAN se debe usar el 4789 por defecto [31]. En cuanto a los identificadores de las VxLANs se usan los mismos en los túneles de ida y de vuelta y deben ser distintos en el mismo sdedge. La dirección remota del túnel indica donde debe acabar este, como se observa esta IP sí que es una pública en el escenario. Por último, se activan las interfaces con "up".

En el esquema anterior solo se enseña una representación del túnel y no el camino real. Este para por la red de nivel 3 a través de los dispositivos de traducción de direcciones y después va a la red del ISP acabando el túnel de nuevo en la NAT de la otra corporación la figura siguiente enseña la representación del camino real.

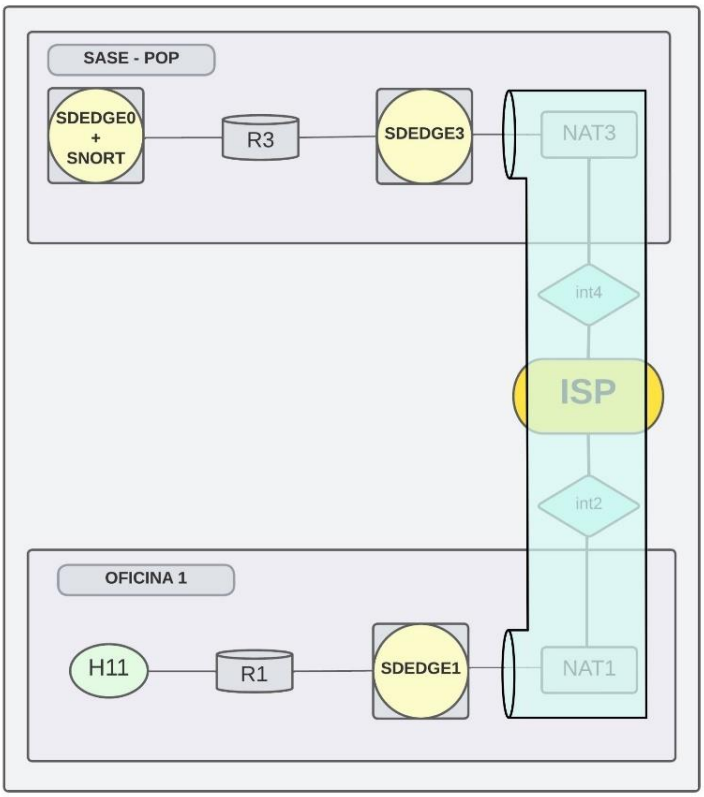


Figura 31: Túnel real entre SASE-POP y oficina 1

- Línea 32: *add-port br0 vxlan2*
- Línea 33: *add-port br0 eth1*
- Línea 34: *add-port br0 eth3*
- Línea 35: *add-port br0 veth0*
- Línea 36: *add-port br1 veth1*

La última parte del código que se ejecuta al arrancar los sdedges es añadir los puertos a los distintos bridges, a continuación, se muestra un esquema con la situación final haciendo uso del comando <<ovs-vsctl show>>.

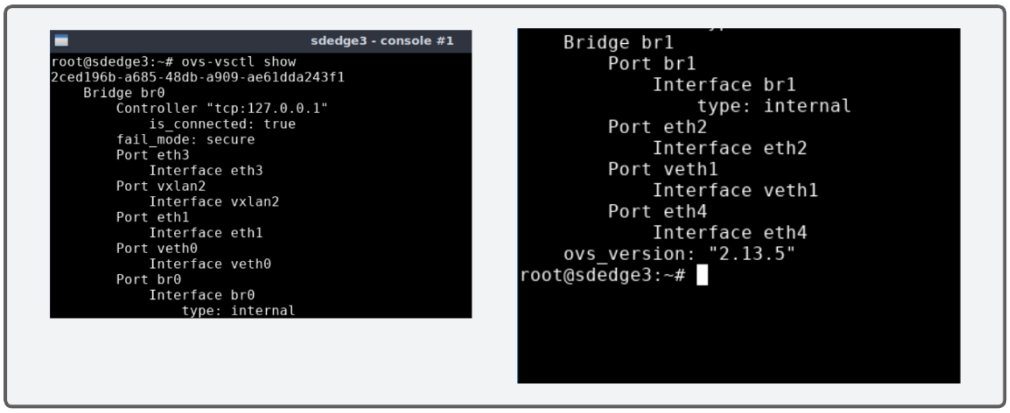


Figura 32: Configuración bridges SDEDGE3

Estas capturas muestran la configuración de cada bridge. En la siguiente figura se va a tratar de simplificar para comprender el funcionamiento.

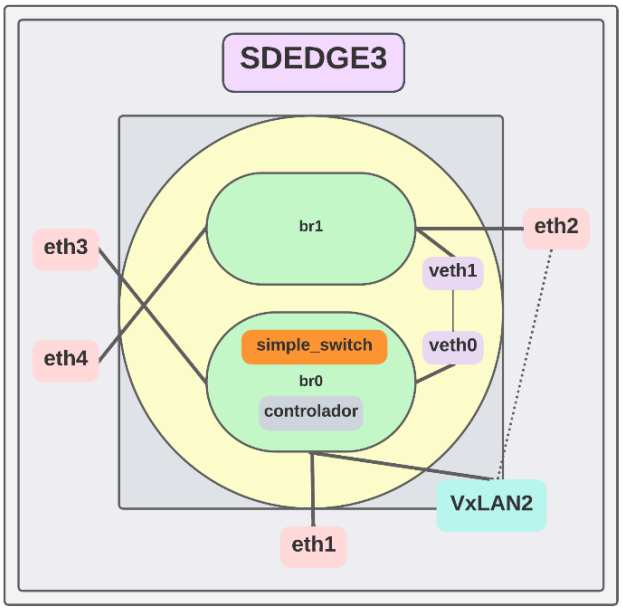


Figura 33: Mapa puertos SDEDGE3

La configuración de los puertos dentro del switch, dividido en 2 bridges, es la que se observa. El propósito es la encapsulación del tráfico para los túneles VxLAN, aunque a veces también se usan estos con un propósito de separación de tráfico dentro de un switch. Se observa como el simple_switch se instala en br0. Esta figura es un tanto complicada debido a dos puntos principales que son las interfaces virtuales (veth) y el túnel VxLAN2.

El switch instalado en br0 es el encargado de redirigir el tráfico por los distintos puertos. La interfaz de VxLAN2 es el punto donde estará el VTEP y por consecuencia donde se encapsula el tráfico. Esta interfaz luego se reenvía a través de eth2 con las nuevas cabeceras añadidas. OVS se encarga internamente de esta comunicación entre br0 y br1, en la Figura 33 se ha marcado entre puntos ese enlace, el cual pasa por br1 internamente.

Como demostración para entender la figura se va a analizar el tráfico en dos situaciones:

1. Tráfico a internet:

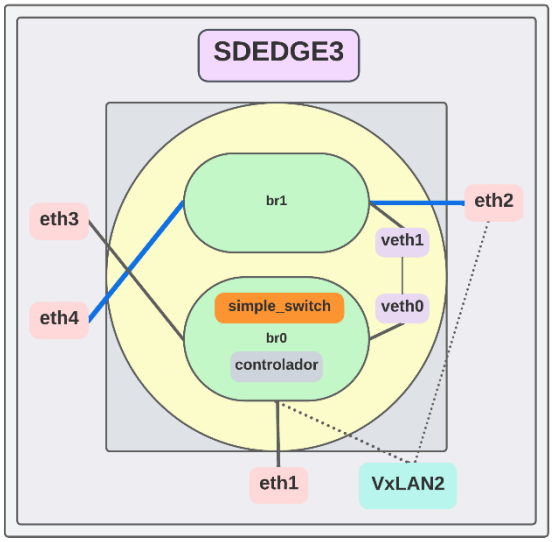


Figura 34: Ruta desde h01 a internet

Este camino simplemente usa el br1, que interconecta eth4 con eth2.

2. Tráfico a la oficina 1 usando VxLAN:

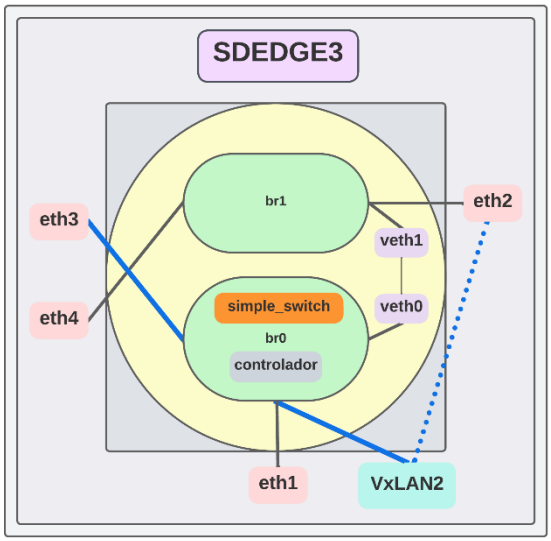


Figura 35: Ruta desde h01 a la oficina 1

La Figura 35 muestra cómo se encaminan los paquetes para llegar a la oficina 1 desde h01. Algunas de las principales observaciones son como se deja la antigua interfaz del switch “eth1” la cual está conectada a la WAN entre oficinas para usar la nueva red encapsulando el tráfico.

El punto de interés de este análisis reside en la interfaz VxLAN2 donde se sitúa el VTEP el cual encapsula el tráfico para que sea reenviado.

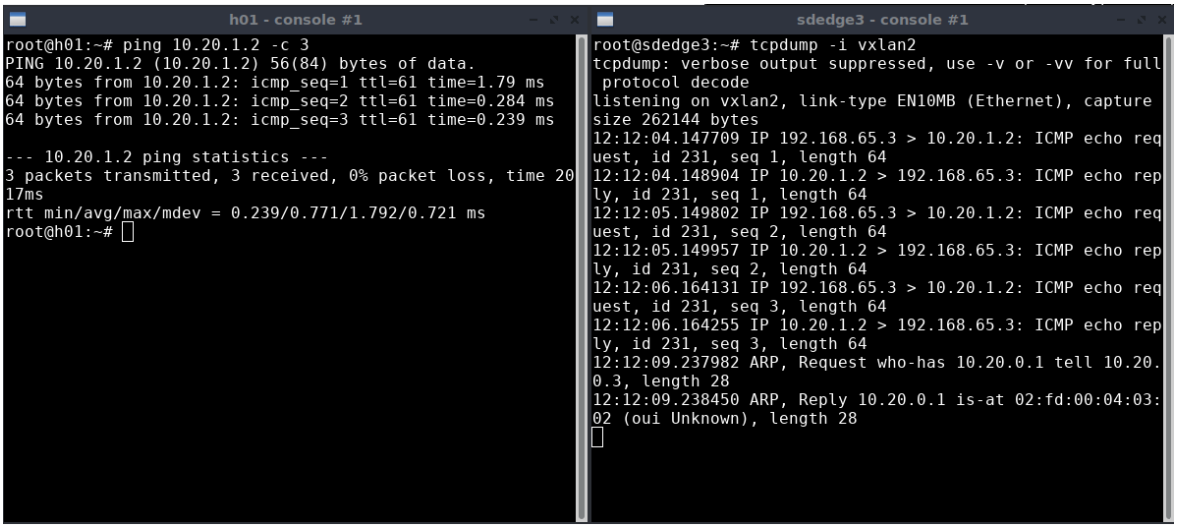


Figura 36: Análisis en interfaz VxLAN2

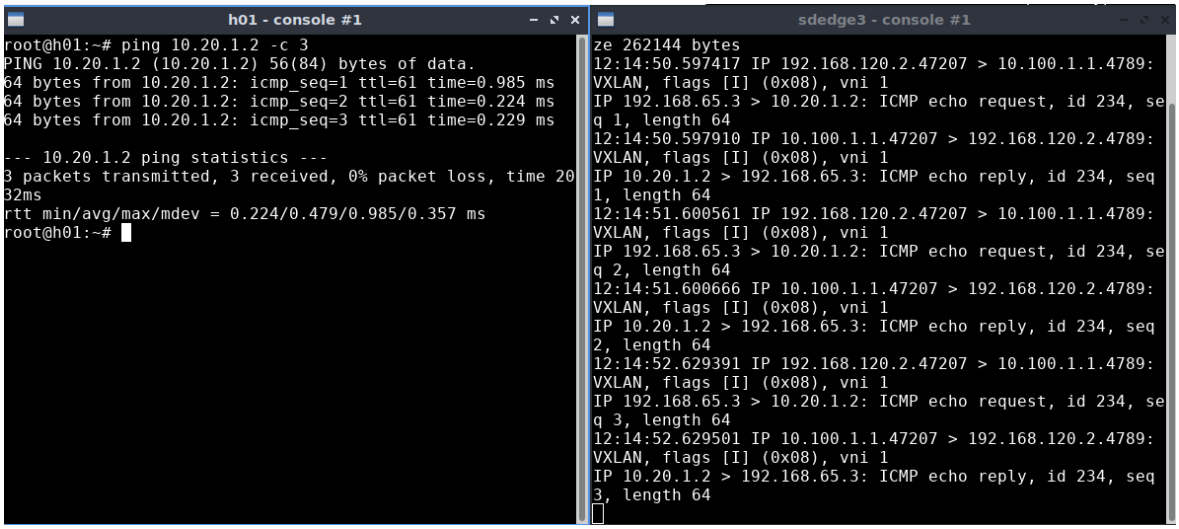


Figura 37: Análisis en interfaz eth2

La primera captura muestra como el tráfico entre por la interfaz VxLAN2 sin ningún tipo de indicador sobre el túnel. En cuanto a la segunda captura se observa como ya aparece el identificador VNI junto con la cabecera VxLAN y además se indican las IPs que se han añadido a la cabecera y las originales que se encuentran encapsuladas como se muestra en la Figura 6: Ejemplo encapsulación de VxLAN en VTEP.

Se va a analizar a continuación de manera detallada el funcionamiento del SDEDGE3 indicando con ejemplos las rutas que siguen los paquetes que llegan al dispositivo.

Primero se deben identificar los puertos del SDEDGE3, en concreto del bridge en el que se encuentra el controlador el cual en nuestro caso es el 0. Tenemos distintas opciones para ver el número de puerto asociado a cada interfaz como por medio de la consola haciendo uso de <<ovs-ofctl show br0>> o directamente desde la interfaz gráfica accediendo desde el navegador.

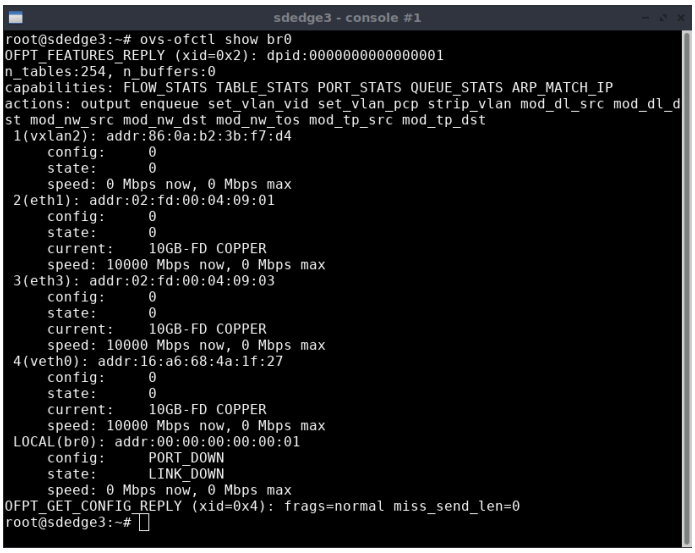


Figura 38: Puertos numerados SDEDGE3

Una vez vistos los puertos del bridge 0, pasamos a observar los flujos que se han creado gracias al controlador en el SDEDGE. De nuevo estos flujos se pueden observar con el comando <<ovs-ofctl dump-flows br0>> aunque en este caso por comodidad se observan directamente desde el navegador.

	PRIORITY	MATCH FIELDS	COOKIE	DURATION	IDLE TIMEOUT	HARD TIMEOUT	INSTRUCTIONS	PACKET COUNT	BYTE COUNT	FLAGS
☐	1	in_port = 3 eth_src = 02:fd:00:04:05:02 eth_dst = 02:fd:00:04:03:02	0	31163	0	0	OUTPUT:1	27	2366	0
☐	1	in_port = 1 eth_src = 02:fd:00:04:03:02 eth_dst = 02:fd:00:04:05:02	0	31163	0	0	OUTPUT:3	26	2268	0
☐	1	in_port = 4 eth_src = 02:fd:00:04:0d:01 eth_dst = 9e:ed:a6:8c:95:4e	0	26875	0	0	OUTPUT:4	38	6269	0
☐	1	in_port = 4 eth_src = 9e:ed:a6:8c:95:4e eth_dst = 02:fd:00:04:0d:01	0	25695	0	0	OUTPUT:4	13	1042	0
☐	1	in_port = 2 eth_src = 02:fd:00:04:0c:01 eth_dst = 5e:af:21:92:4b:44	0	25690	0	0	OUTPUT:2	10	420	0
☐	1	in_port = 4 eth_src = 02:fd:00:04:0d:01 eth_dst = 02:fd:00:04:03:03	0	7234	0	0	OUTPUT:2	0	0	0
☐	0	ANY	0	31194	0	0	OUTPUT:CONTROLLER	181	12460	0

Figura 39: Flujos de SDEDGE3

Se comentan de forma resumida los campos que encontramos en la tabla:

- Priority: para que el controlador decida en caso de que más de una entrada encaje con el paquete entrante, a mayor número mayor prioridad.

- Match Fields: campos que deben encajar para decir mandar el paquete por un puerto.
- Cookie: permite que el controlador de un identificador a los flujos.
- Duration: tiempo activo de cada entrada.
- Idle Timeout: tiempo sin recibir un paquete para que se elimine la entrada.
- Hard Timeout: tiempo para eliminar una entrada, aunque se sigan recibiendo paquetes.
- Instructions: que debe hacer el controlador con los paquetes que encajan, normalmente sacarlos por otro puerto.
- Packet Count: número de paquetes de cada fila.
- Byte count: número de bytes de cada fila.
- Flags: permiten alterar como se maneja una entrada [32].

Se va a comprobar el camino que debe seguir un paquete que va desde el dispositivo del trabajador hasta la red corporativa.

Primero se comprueban las direcciones MAC del salto anterior al SDEDGE3 en este caso R3. Se exploran las interfaces del dispositivo con <<ifconfig>> y se observa que la segunda interfaz, eth2, por la cual se deberían mandar los paquetes hacia h11 tiene la dirección MAC: 02:fd:00:04:05:02. El segundo paso es buscar la dirección destino de los paquetes que pasan por este SDEDGE, como se mostró al analizar la red de la oficina 1 se marca como destino el router de la oficina destino, en este caso R1, la MAC de la interfaz deseada es: 02:fd:00:04:03:02. Como último paso antes de verificar los flujos falta comprobar el puerto de entrada.

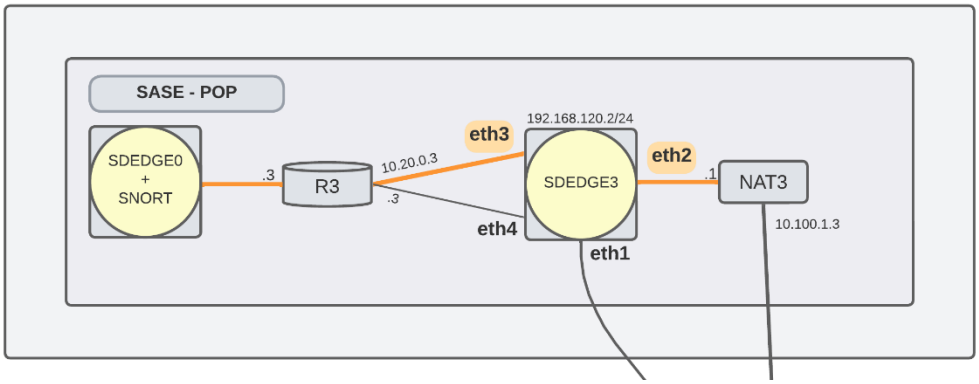


Figura 40: Ruta hacia RC1

Como hemos visto antes eth3, puerto que pertenece al bridge 0, es el número 3 y aunque parece que debe salir por eth2[5], debido a que hemos creado un túnel con una nueva interfaz vxlan2 es por donde sale y si se comprueba se confirma que es el número 1. En este punto se recuerdan todos los valores que tenemos:

- MAC-origen: 02:fd:00:04:05:02
- MAC-destino: 02:fd:00:04:03:02
- Puerto entrante: 3

[5] De manera física sí que se utiliza la interfaz eth2. Se puede comprobar mediante <<tcpdump -i eth2>> mientras se manda tráfico hacia cualquiera de las redes.

Se busca una fila que encaje con estos campos. En realidad, se van a encontrar dos caminos, uno de ida y otro de vuelta.

PRIORITY	MATCH FIELDS	COOKIE	DURATION	IDLE TIMEOUT	HARD TIMEOUT	INSTRUCTIONS	PACKET COUNT	BYTE COUNT	FLAGS
1	in_port = 3 eth_src = 02:fd:00:04:05:02 eth_dst = 02:fd:00:04:03:02	0	36084	0	0	OUTPUT:1	1619	153118	0
1	in_port = 1 eth_src = 02:fd:00:04:03:02 eth_dst = 02:fd:00:04:05:02	0	36084	0	0	OUTPUT:3	1618	153020	0

Figura 41: Demostración flujos-1

PRIORITY	MATCH FIELDS	COOKIE	DURATION	IDLE TIMEOUT	HARD TIMEOUT	INSTRUCTIONS	PACKET COUNT	BYTE COUNT	FLAGS
1	in_port = 3 eth_src = 02:fd:00:04:05:02 eth_dst = 02:fd:00:04:03:02	0	36136	0	0	OUTPUT:1	1672	158200	0
1	in_port = 1 eth_src = 02:fd:00:04:03:02 eth_dst = 02:fd:00:04:05:02	0	36136	0	0	OUTPUT:3	1671	158102	0

Figura 42: Demostración flujos-2

La primera fila corresponde al camino de ida y la segunda entrada al de vuelta, para hacer la comprobación se mandan paquetes de forma continua, por ejemplo, con un ping y se mira si aumenta el número de paquetes en la antepenúltima columna. Como se observa efectivamente aumentan ambas filas, ya que por cada mensaje enviado se recibe otro.

CONFIGURACIÓN NAT3

La función principal de la NAT (Network Address Translation) es la conocida originalmente, traducir direcciones IP privadas en públicas y viceversa.

En el caso del escenario, el dispositivo se encuentra situado en el borde de las corporaciones antes de acceder al ISP. Tiene una interfaz que conecta directamente con el SDEDGE correspondiente y otra con una IP pública que conecta con el ISP, en el caso de la NAT3 es por medio del bridge virtual int4.

```
0.
1.<exec seq="on_boot" type="verbatim">
2.  /usr/bin/vnx_config_nat eth1 eth2
3.  iptables -t nat -A PREROUTING -p udp -d 10.100.1.3 --dport 4789 -j DNAT --to-
          destination 192.168.120.2
4.  iptables -t nat -A PREROUTING -p tcp -d 10.100.1.3 --dport 8080 -j DNAT --to-
          destination 192.168.120.2
5.</exec>
6.
```

Figura 43: Extracto código configuración NAT3

La configuración de NAT3 tiene dos partes principales:

- Script vnx_config_nat: se hace uso del script del paquete de VNX, desarrollado por David Fernández. La función principal de este script es configurar la NAT entre dos interfaces. Por ello cuando se llama al script se le pasa como argumento las interfaces eth1 y eth2.

```
33  USAGE="
34  Usage:
35
36  vnx_config_nat [-d] <internal_if> <external_if>
37
38      being:
39          <internal_if>: the name of the internal interface.
40          <external_if>: the name of the public interface.
41          [-d]:          use this option to delete the NAT rules
42  "
```

Figura 44: Uso script NAT VNX

- Añadir reglas a la NAT creada: se añaden dos reglas de NAT con la misma sintaxis. “iptables -t nat -A PREROUTING” es la parte inicial para indicar que se va a añadir una nueva regla “-p udp -d 10.100.1.3 --dport 4789” se indica el protocolo, la IP y el puerto destino y después “-j DNAT --to-destination 192.168.120.2” a donde se debe enviar si un paquete coincide con los campos anteriores. Se exponen las reglas en el siguiente cuadro:

Original			Traducción NAT
Protocolo	**Puerto**	**IP Destino**	**IP Destino**
TCP	8080	10.100.1.3	192.168.120.2
UDP	4789	10.100.1.3	192.168.120.2

Figura 45: Reglas NAT

- Las reglas se definen para reencaminar el tráfico al interior del SASE-POP. En concreto la de puerto 4789 se usa para la VxLAN y EL 8080 para, como se explica a continuación, acceder a la interfaz gráfica del controlador Ryu.

4.2.4. CONFIGURACIÓN DE LA OFICINA 1:

La Figura 10 muestra como las modificaciones principales de las otras oficinas residen en los SDEDGE. La dificultad de integrar dos túneles en la oficina 1 se soluciona con los identificadores VNI y con una forma distinta de declarar los túneles en el VNX. En este apartado se va a tratar el desarrollo sobre los scripts de SDEDGE1 y los cambios necesarios.

DESARROLLO SCRIPTS SDEDGE1

Como última parte del desarrollo se definen los scripts que se han utilizado en SDEDGE1. El script en cuestión es una modificación del simple_switch_13.py, un script bajo la licencia de Apache. La aplicación de simple_switch trabaja con el controlador de ryu y permite de forma resumida hacer de switch en el SDEDGE. Como información adicional se comenta que la sintaxis de estos archivos en cuanto al número final es decir "13" hace referencia a la versión de OpenFlow en este caso la 1.3.

En el momento en el que se arranca el escenario, el SDEDGE solo cuenta con un controlador. El controlador no dispone de instrucciones de procesamiento al recibir las peticiones ARP, por lo que no es capaz de entender y crear los flujos necesarios para actuar como switch. Se podría decir que la red SD-WAN son estos scripts que se arrancan desde los SDEDGEs. Es importante recalcar que el script se conecta con el controlador es decir con el bridge 0, br0.

De manera resumida este script hace que cuando se recibe un paquete en el SDEDGE se llame a una función que maneja los "packet_in" de ovs. Después se extrae la información relevante del paquete que son las MACs y el puerto entrante. A continuación, busca en su diccionario si encaja con alguna MAC destino para reenviarlo por un puerto y crear el flujo, sino es así se inundará basando en unas VLANs definidas.

Ryu cuenta con distintas aplicaciones en este caso la que se va a usar es la de un switch con alguna modificación. Este código tendrá las distintas instrucciones genéricas de inundar al no saber el puerto en el que está el destino y de aprender y ser capaz de generar los flujos que se enseñan a continuación. Debido a que en el SDEDGE de la oficina 1 se tiene una interfaz más que en las demás, por el túnel VxLAN añadido, se necesita modificar el script.

Una vez recordada esta imagen se puede entender de manera más sencilla las modificaciones que se hacen al código original. El cual se configuraba con unas VLANs para separar los puertos que encaminan tráfico entre redes y los que llevan el resto del tráfico.

```
332.
333.portToVLANDict = {1:500, 2:500, 3: None, 4:500, 5: None}
334.
```

Figura 46: Modificación Script simple_switch

```
332.
333.self.mac_to_port[dpid][src] = in_port
334.print(self.mac_to_port)
335.if          (dst          in          self.mac_to_port[dpid]          and
   portToVLANDict[self.mac_to_port[dpid][dst]] ==  portToVLANDict[in_port]):
336.    out_ports = [self.mac_to_port[dpid][dst]]
337.    newFlow = True
338.else:
339.    out_ports  =  [k  for  k,  v  in  portToVLANDict.items()  if  v  ==
   portToVLANDict[in_port] and k != in_port]
340.    newFlow = False
341.
```

Figura 47: Uso VLAN para inundar y crear flujos

Como última mención para no tener que escribir el comando entero de ejecución, es decir por simplificar el escenario, se incluye un archivo de tipo shell (.sh) con el código que se ejecuta en este simple_switchs.

```
335.
336.#!/bin/bash
337.ryu-manager     /root/flowmanager/flowmanager.py     ~/simple_switch_14.py
    ryu.app.ofctl_rest
338.
```

Figura 48: Código ryu51.sh

Una gran ventaja de las nuevas versiones de openFlow son sus interfaces gráficas disponibles para poder visualizar y manejar el mismo desde ahí, obviamente se podría hacer desde la propia consola. Para poder acceder a estas se debe introducir en un navegador su IP, en este caso la de la NAT1 con el puerto 8080, el cual como dijimos antes llevará al SDEDGE en el momento de traducción de direcciones.

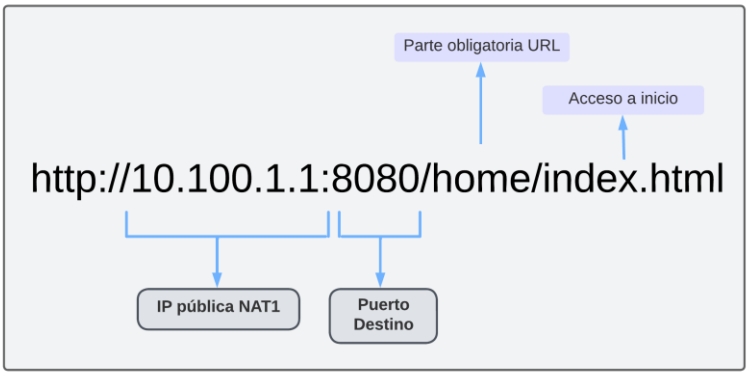


Figura 49: Acceso GUI Controlador

5. RESULTADOS

Desde el principio, el objetivo de este TFG ha sido demostrar el funcionamiento de una prueba de concepto de SASE. En este capítulo se van a describir los resultados de las pruebas de funcionamiento que se han realizado para validar el desarrollo. Para obtener los resultados se han utilizado las siguientes herramientas:

- Tcpdump: es una herramienta de línea de comandos incluida en los sistemas operativos Linux. Tcpdump permite mediante el argumento "-i" seleccionar una interfaz y observar todo el tráfico que circula por el mismo.

- Wireshark: es una herramienta de análisis de paquetes. Wireshark permite ver el contenido de cada paquete como las cabeceras de protocolos, direcciones IP o puertos. Esta herramienta contiene complementos para el seguimiento de paquetes o etiquetado de los mismos.

- Snort: se mostrarán las estadísticas y alertas de la herramienta instalada. Estos resultados aparecerán por la consola en la que se incorpora el IDS.

Existen distintas formas de generar tráfico para probar el escenario desarrollado. En la mayoría de las pruebas se utiliza simplemente un "ping" entre las máquinas para comprobar que existe la conexión.

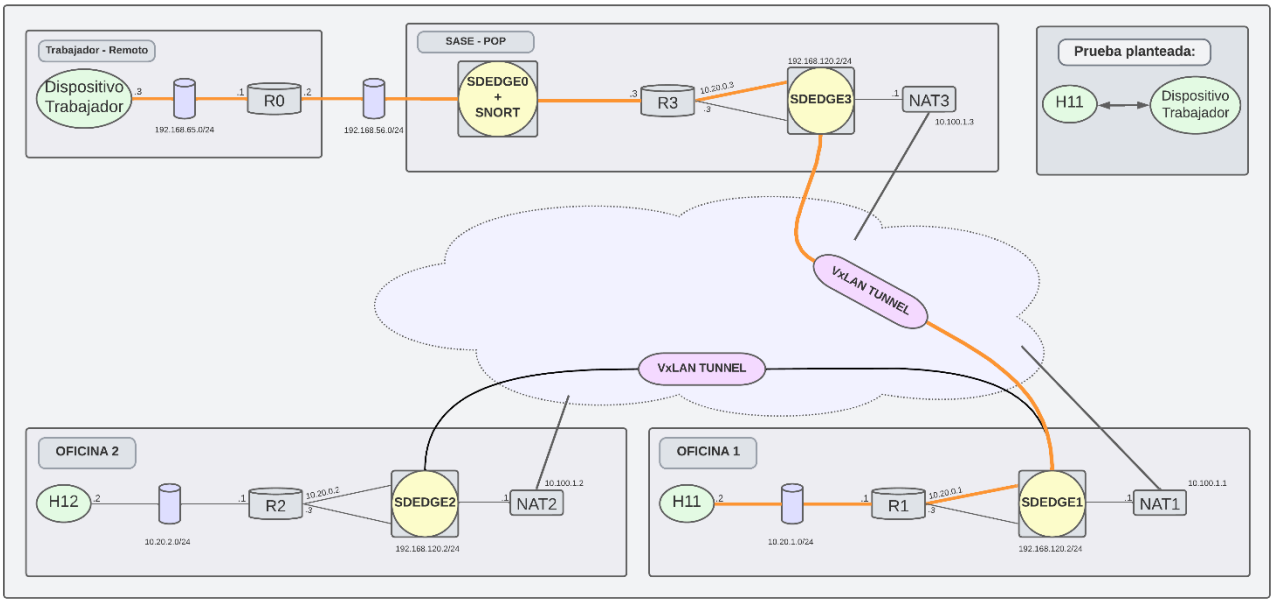


Figura 50: Pruebas planteadas sobre el escenario

Aunque este escenario permite hacer distintas pruebas, el interés real reside en la interconexión de estos dos puntos. Por esto se analiza a fondo esta conexión y no se profundiza en otras partes del escenario.

5.1. CONEXIÓN ENTRE H11 Y DISPOSITIVO DEL TRABAJADOR

Para estudiar los resultados de esta prueba se empieza por realizar un ping entre las dos máquinas, por ejemplo, desde h11 hasta el dispositivo del trabajador. La idea principal es comprobar que existe conexión entre ambos. Usamos: `ping 192.168.65.3`

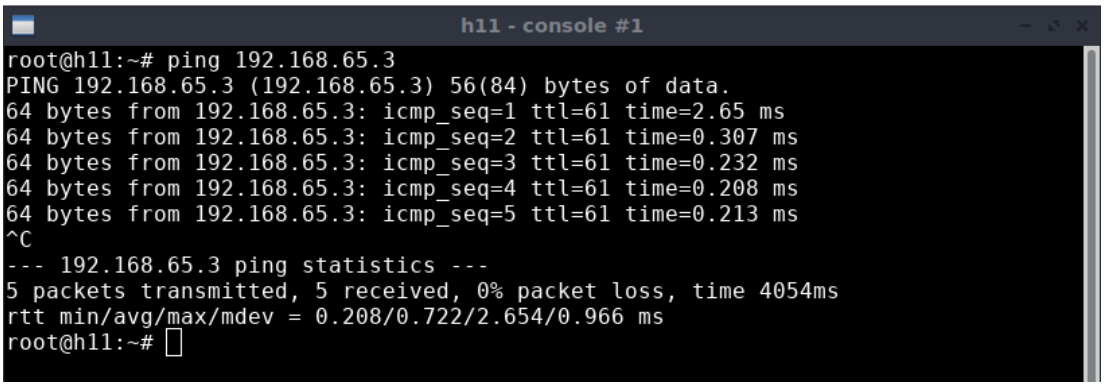


Figura 51: Ping entre h11 y trabajador remoto

En esta captura se puede apreciar como la conexión entre h11 y el dispositivo del trabajador remoto esta activa. Una de las principales observaciones a tener en cuenta es el tiempo de la primera secuencia del ICMP. Las causas principales de este tiempo tan elevado son dos, por una parte la instalación de cada entrada en los flujos de los sdedge y la otra el establecimiento del túnel vxlan.

En este punto del análisis es importante recordar la Figura 31. En esa figura se recuerda que los túneles VxLAN reales pasan por los dispositivos NAT. Esta observación es muy importante ya que gran parte del análisis se va a basar en los dispositivos NAT. Debido al interes creado por el túnel vxlan se va a analizar el paquete antes de entrar al mismo, en el interior y al salir. El análisis se va a hacer sobre el paquete de ida, aunque podría hacerse sobre los de vuelta de igual forma.

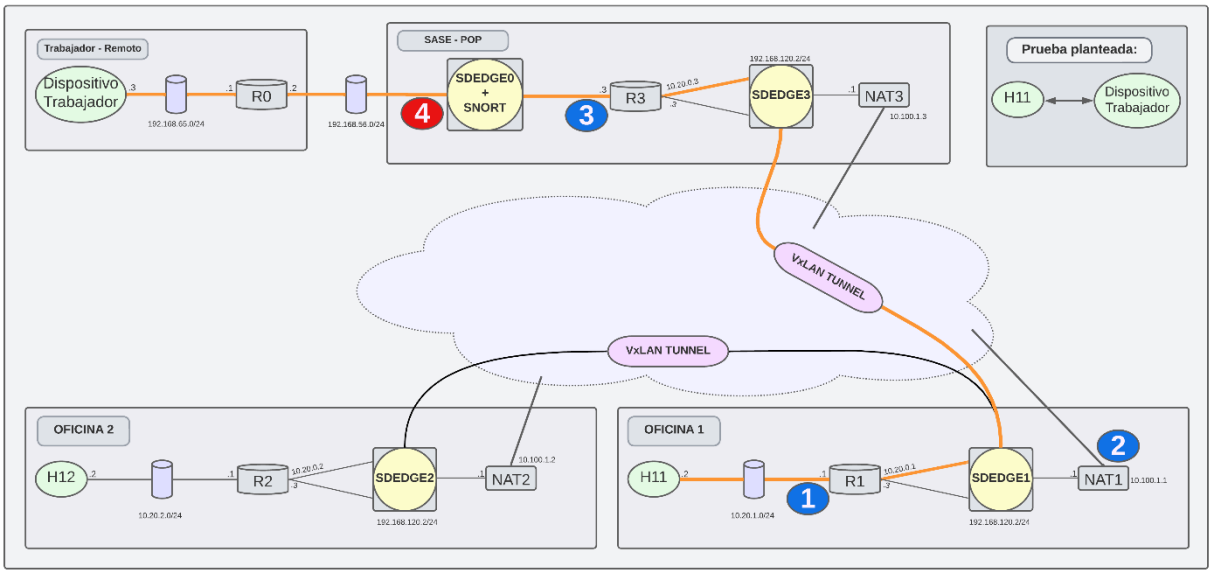
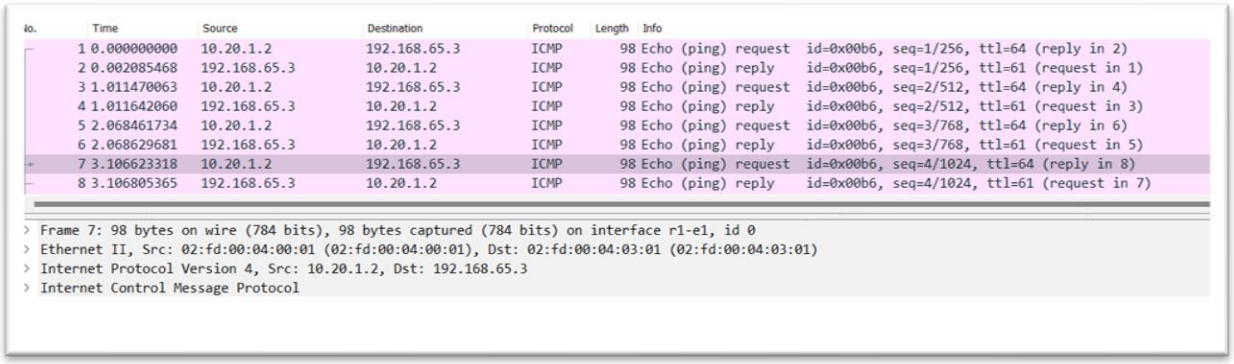


Figura 52: Puntos de análisis de tráfico

En esta figura se muestran los puntos de análisis del tráfico. En el punto 4 se haran las pruebas sobre la seguridad de la red con el IDS instalado.

1. Interfaz 1 – R1:



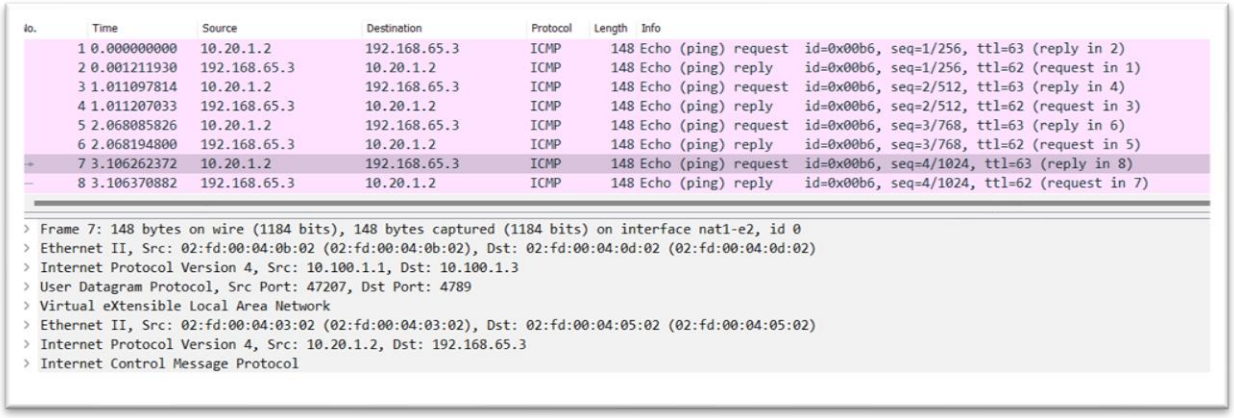
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	10.20.1.2	192.168.65.3	ICMP	98	Echo (ping) request id=0x00b6, seq=1/256, ttl=64 (reply in 2)
2	0.002085468	192.168.65.3	10.20.1.2	ICMP	98	Echo (ping) reply id=0x00b6, seq=1/256, ttl=61 (request in 1)
3	1.011470063	10.20.1.2	192.168.65.3	ICMP	98	Echo (ping) request id=0x00b6, seq=2/512, ttl=64 (reply in 4)
4	1.011642060	192.168.65.3	10.20.1.2	ICMP	98	Echo (ping) reply id=0x00b6, seq=2/512, ttl=61 (request in 3)
5	2.068461734	10.20.1.2	192.168.65.3	ICMP	98	Echo (ping) request id=0x00b6, seq=3/768, ttl=64 (reply in 6)
6	2.068629681	192.168.65.3	10.20.1.2	ICMP	98	Echo (ping) reply id=0x00b6, seq=3/768, ttl=61 (request in 5)
7	3.106623318	10.20.1.2	192.168.65.3	ICMP	98	Echo (ping) request id=0x00b6, seq=4/1024, ttl=64 (reply in 8)
8	3.106805365	192.168.65.3	10.20.1.2	ICMP	98	Echo (ping) reply id=0x00b6, seq=4/1024, ttl=61 (request in 7)

```
> Frame 7: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface r1-e1, id 0
> Ethernet II, Src: 02:fd:00:04:00:01 (02:fd:00:04:00:01), Dst: 02:fd:00:04:03:01 (02:fd:00:04:03:01)
> Internet Protocol Version 4, Src: 10.20.1.2, Dst: 192.168.65.3
> Internet Control Message Protocol
```

Figura 53: Wireshark captura en R1

En esta primera captura se elige un paquete el cual se tratará de seguir por la red en las proximas capturas, una forma de conseguir esto es mediante el identificador de ICMP. En este caso no obtenemos demasiada información ya que simplementes se trata de un ping con las cabeceras y campos esperados. Por destacar vemos como la IP destino aunque sea privada la red corporativa esta preparada para llegar hasta la misma.

2. Interfaz 2 – NAT1:



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	10.20.1.2	192.168.65.3	ICMP	148	Echo (ping) request id=0x00b6, seq=1/256, ttl=63 (reply in 2)
2	0.001211930	192.168.65.3	10.20.1.2	ICMP	148	Echo (ping) reply id=0x00b6, seq=1/256, ttl=62 (request in 1)
3	1.011097814	10.20.1.2	192.168.65.3	ICMP	148	Echo (ping) request id=0x00b6, seq=2/512, ttl=63 (reply in 4)
4	1.011207033	192.168.65.3	10.20.1.2	ICMP	148	Echo (ping) reply id=0x00b6, seq=2/512, ttl=62 (request in 3)
5	2.068085826	10.20.1.2	192.168.65.3	ICMP	148	Echo (ping) request id=0x00b6, seq=3/768, ttl=63 (reply in 6)
6	2.068194800	192.168.65.3	10.20.1.2	ICMP	148	Echo (ping) reply id=0x00b6, seq=3/768, ttl=62 (request in 5)
7	3.106262372	10.20.1.2	192.168.65.3	ICMP	148	Echo (ping) request id=0x00b6, seq=4/1024, ttl=63 (reply in 8)
8	3.106370882	192.168.65.3	10.20.1.2	ICMP	148	Echo (ping) reply id=0x00b6, seq=4/1024, ttl=62 (request in 7)

```
> Frame 7: 148 bytes on wire (1184 bits), 148 bytes captured (1184 bits) on interface nat1-e2, id 0
> Ethernet II, Src: 02:fd:00:04:0b:02 (02:fd:00:04:0b:02), Dst: 02:fd:00:04:0d:02 (02:fd:00:04:0d:02)
> Internet Protocol Version 4, Src: 10.100.1.1, Dst: 10.100.1.3
> User Datagram Protocol, Src Port: 47207, Dst Port: 4789
> Virtual eXtensible Local Area Network
> Ethernet II, Src: 02:fd:00:04:03:02 (02:fd:00:04:03:02), Dst: 02:fd:00:04:05:02 (02:fd:00:04:05:02)
> Internet Protocol Version 4, Src: 10.20.1.2, Dst: 192.168.65.3
> Internet Control Message Protocol
```

Figura 54: Wireshark captura en NAT1 - 1

Este punto de análisis resulta de gran interes. Se puede observar como la cabecera IP y Ethernet se han duplicado, además aparece la cabecera VxLAN y la de UDP. En este paso se recuerda la encapsulación que se da en VxLAN la Figura 6. A continuación se muestra de manera detallada las cabeceras para un análisis en profundidad.

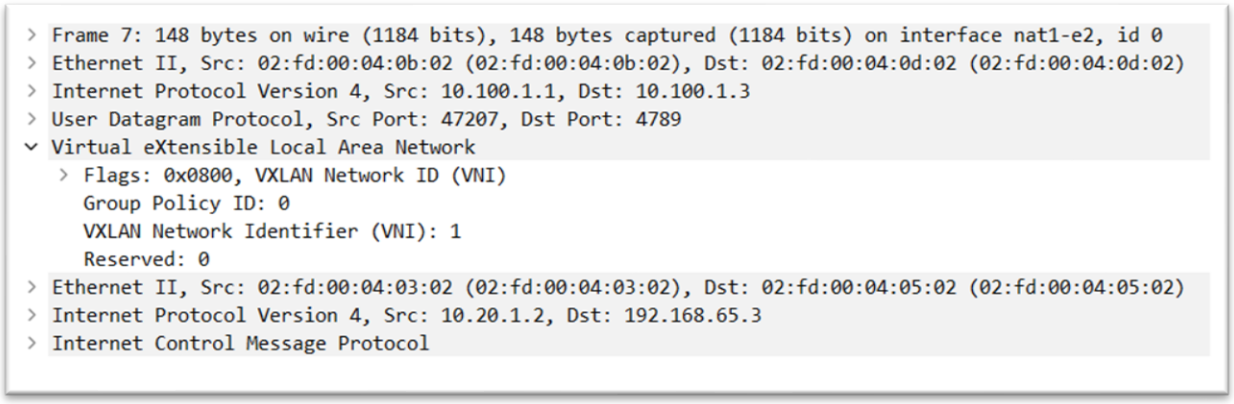


Figura 55: Wireshark captura en NAT1 - 2

En cuanto a la cabecera VxLAN, Wireshark permite ver el identificador del túnel. Se recuerda que en la Figura 27, se eligió el "1", línea 30, como identificador del túnel.

Las cabeceras exteriores que envuelven al paquete original contienen a nivel IP las direcciones que se han descrito en el momento de declarar el túnel. El destino es la IP 10.100.1.3 es decir NAT3, como origen se tiene la dirección pública de la NAT1. En cuanto al puerto destino se puede observar que en la cabecera UDP se marca el 4789. Las cabeceras que quedan dentro del envoltorio de VxLAN son las originales, cambiando las direcciones MAC por las del salto anterior. Se puede comprobar la similitud al comparar con la Figura 53: Wireshark captura en R1.

En el momento en el que el paquete se recibe en el final del túnel, SDEDGE3, se eliminan todas las cabeceras que no se encuentran encapsuladas por la cabecera VxLAN. Es importante recordar que en las NAT se tiene una regla para traducir direcciones. De la pública de cada NAT con puerto 4789 a la dirección IP privada del SDEDGE correspondiente.

Un punto relevante a observar es como el mismo paquete aumenta de tamaño de forma considerable solo por la encapsulación de las cabeceras. En la Figura 53: Wireshark captura en R1se observa en la primera fila el tamaño de 98 Bytes y viendo el mismo paquete en la Figura 55: Wireshark captura en NAT1 - 2, el tamaño aumenta hasta los 148 Bytes. Es decir, se aumenta en 50 Bytes el tamaño, por ello se tuvo que reducir la MTU en el desarrollo del escenario en las máquinas previas al túnel.

3. Interfaz 1 - R3:

Figura 56: Wireshark captura en R3

En este último punto de análisis del tráfico en R3, se observa como las cabeceras exteriores del punto anterior se han eliminado y se vuelve a tener el paquete original. Si se omite la captura intermedia parece que el trabajador remoto se encuentra en la misma red de la oficina 1.

4. Snort:

Se va a comprobar que las alertas creadas funcionan correctamente. Este es el punto que añade seguridad a la red y en el que se configuran las alertas necesarias. Para el ping desde h11 se puede observar por pantalla en siguiente contenido:

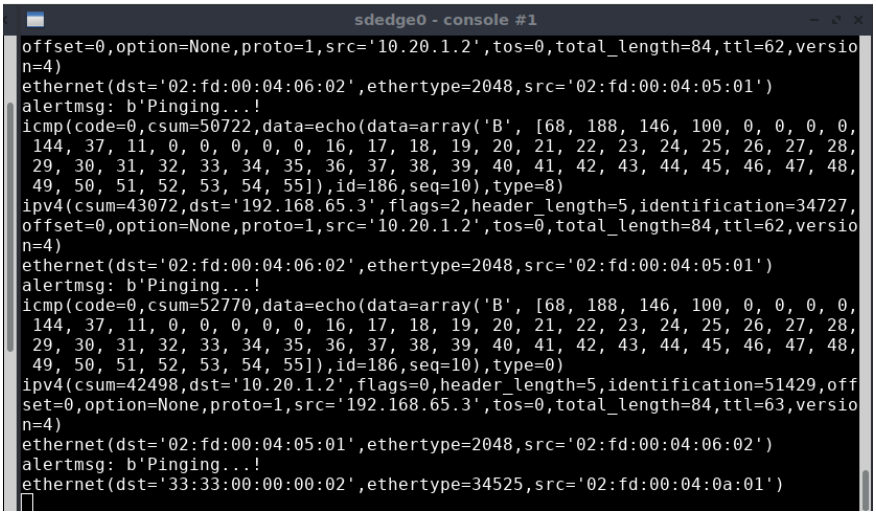


Figura 57: Snort regla Ping

Si se mira de manera detallada cada mensaje se puede observar cómo realmente se generan estos mensajes dos veces ya que el ping es de ida y vuelta, por ello las MACs destino y origen van cambiando.

Al haber incluido una regla general para hacer ping, esta salta de manera constante cuando hay tráfico. Esta regla solo es útil para comprobar el funcionamiento del IDS ya que no aporta información relevante. Por ello en las siguientes comprobaciones se van a eliminar algunas alertas para permitir ver con claridad las alertas SCP.

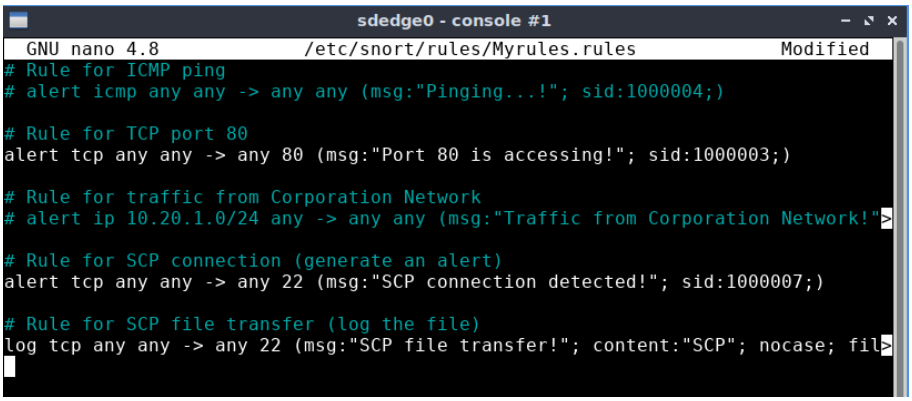


Figura 58: Modificación Myrules.rules

Se va a usar SCP para comprobar la última alerta programada. SCP es un protocolo para copiar archivos desde o hasta otro dispositivo de remoto. La prueba consiste en copiar un archivo a h11 desde el dispositivo del trabajador remoto. Para ello se crea un archivo en h01 llamado scp.prueba y a continuación se trata de enviar a h11 con el comando: `scp scp.prueba root@10.20.1.2:/root`

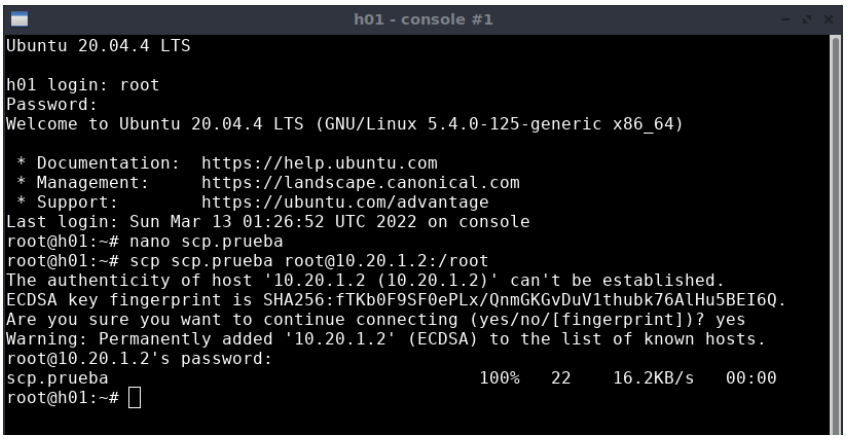


Figura 59: SCP desde h01 a h11

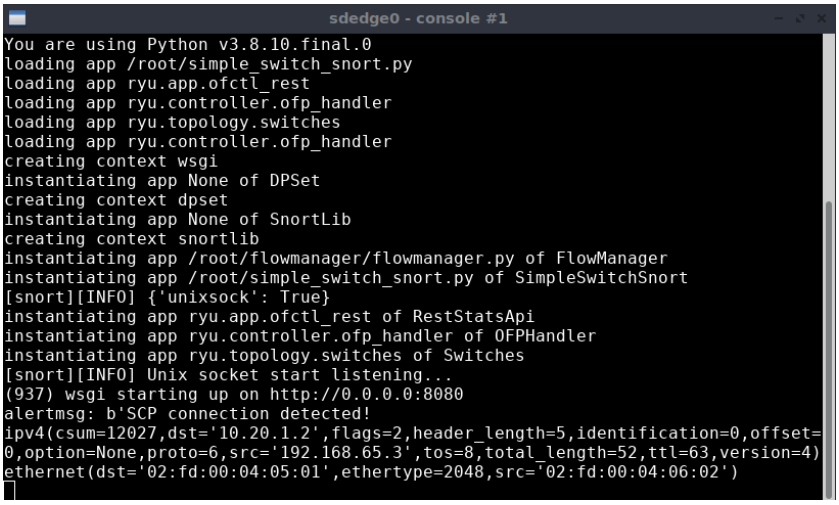


Figura 60: Alerta SCP en SDEDGE0

En estas figuras se puede observar como por una parte se manda correctamente el archivo a h11 y como Snort detecta una conexión y avisa con el mensaje que se indicó en las reglas. En este caso la regla está basada en el uso del puerto 22 como puerto destino, ya que es el que se utiliza en SCP. Por último, se muestra una figura que recoge las estadísticas del tráfico cursado y de las alertas que han saltado en el Snort. Esta es de utilidad a la hora de estudiar el tráfico y entender las alertas.

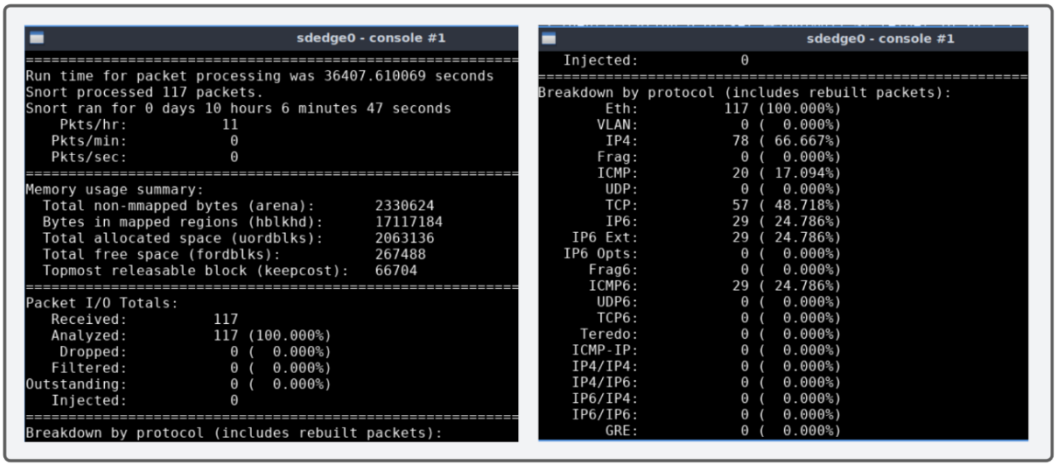


Figura 61: Resumen alertas Snort - 1

Las figuras anteriores recogen las estadísticas del tráfico cursado y de las alertas que han saltado en el Snort. En este caso se muestra una lista por protocolos usados y un resumen general. Esta herramienta es de gran utilidad a la hora de estudiar el tráfico y entender las alertas.

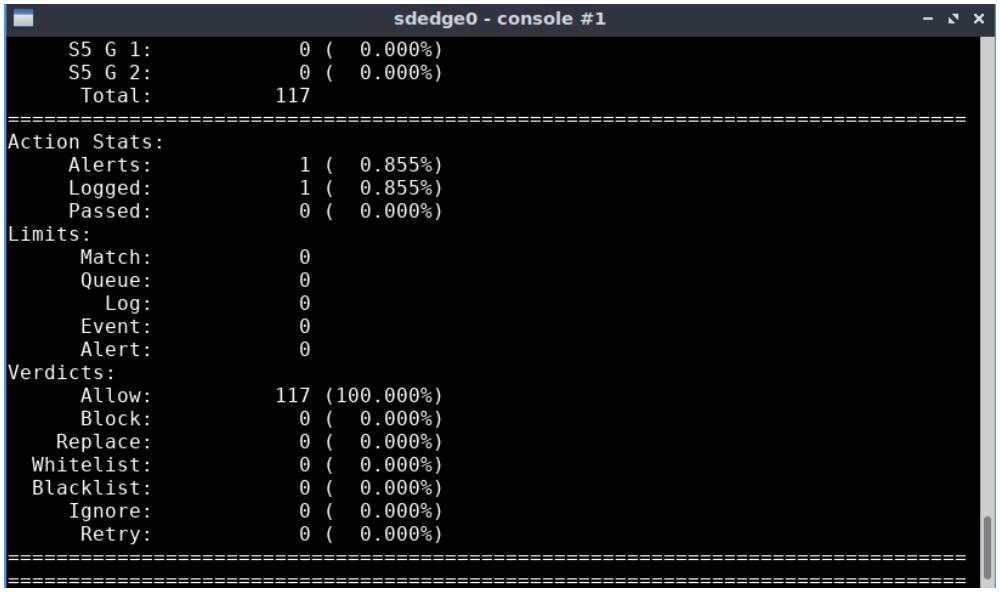


Figura 62: Resumen alertas Snort - 2

Esta última figura es especialmente interesante ya que en ella se muestran las acciones globales realizadas. Se observa como en cuanto al campo de las acciones, se divide en las alertas que han saltado y se registran una alerta. Se aprecia como en el apartado de veredictos se permite el 100% del tráfico.

6. CONCLUSIONES Y LÍNEAS FUTURAS

6.1. CONCLUSIONES

El objetivo de este TFG es desarrollar un escenario para el análisis de servicios SD-WAN con servicios de seguridad basados en el SASE.

Para desarrollar el escenario del TFG, se realizaron varias modificaciones sobre un escenario inicial. En primer lugar, se incorporó la red del trabajador remoto y un SASE-POP. Además, se realizaron ajustes en algunos elementos del escenario base y se llevó a cabo un estudio de soluciones de túneles VxLAN con el objetivo de poder integrar los desarrollos del TFG.

Uno de los puntos a destacar del TFG es la utilización de la herramienta Snort como un sistema de detección de intrusiones en el contexto de las redes definidas por software. La integración de Snort en el SASE-POP tuvo como objetivo demostrar el funcionamiento de la seguridad en la nueva red.

Los resultados obtenidos muestran una exitosa integración de todos los elementos del sistema. Las capturas de pantalla proporcionan una visión clara de cómo las distintas partes del escenario trabajan en conjunto, mostrando la comunicación entre dispositivos de red y la detección y generación de alertas en tiempo real por parte de Snort. En concreto se ha conseguido mostrar la forma de interconexión del trabajador remoto hasta la oficina.

Por último, gracias al desarrollo se ha comprendido como SASE y en concreto SD-WAN permite una simplificación de la infraestructura de red y a su vez de todos los costes asociados a la misma. Además, de las ventajas de un despliegue ágil y flexible, concepto especialmente útil para organizaciones con muchas oficinas. Por otra parte, se ha podido observar el uso de una solución alternativa a las VPNs con VxLAN lo que ha aportado a este trabajo ideas relevantes sobre la encapsulación de cabeceras y la utilidad de estos túneles en el teletrabajo.

En cuanto a la parte de seguridad de SASE. Se observa como la facilidad para añadir seguridad en puntos externos a la corporación resulta sencillo y ventajoso tanto para el cliente como para la corporación. Además, la implementación de las políticas de seguridad en un punto externo a la corporación ofrece una capa de protección adicional.

6.2. LÍNEAS FUTURAS

En cuanto a las posibles líneas futuras para este TFT, se proponen varias mejoras las cuales se pueden dividir en dos líneas de desarrollo:

- Mejora del escenario a nivel de red: este escenario intenta plasmar el concepto de SASE lo mejor posible. Existen algunos aspectos a nivel de red los cuales podrían aportar completitud al escenario.

    - Túnel VxLAN usuario: en cuanto al acceso del usuario al SASE-POP para completar la idea de seguridad y para ser más fiel a la realidad se debería añadir este túnel desde la parte del trabajador remoto al SASE-POP. Esta idea queda representada en el Anexo c: posible ampliación del escenario.

    - SASE-POP: ampliación que pasa por replicar el SASE-POP para ampliar su disponibilidad en caso de que haya alguna caída en estos. Estas estaciones deben estar distribuidas por el mundo por lo que la red que conecta a los mismos podría ser añadida al escenario simulado.

    - Conexiones usuario remoto: en el escenario desarrollado la red del usuario tiene acceso a internet a través del SASE-POP. Se propone desarrollar una mejora la cual permita salir a internet de forma nativa, es decir cuando el usuario no esté trabajando y otra forma de salir mediante el SASE-POP que analice el tráfico cursado y genere alertas cuando el usuario este trabajando.

- Mejora del escenario a nivel de ciberseguridad: debido al ambiente en el que surge y se desarrolla este TFT el apartado de ciberseguridad ha podido quedar más de lado. Por lo que a continuación se proponen mejoras en este aspecto:

    - Firewall de nueva generación (NGFW): el NGFW es una parte indispensable en el despliegue de SASE. Se propone por una parte instalar una solución de firewall tradicional, por ejemplo, en Ryu existe “ryu.app.rest_firewall” o haciendo uso de programas externos dar solución a este problema mediante un NGFW.

    - IDS a IPS: Snort puede funcionar como un sistema de prevención de intrusiones (IPS). Se propone realizar los cambios necesarios para que Snort sea capaz de bloquear tráfico potencialmente peligroso. Existen IDS basados en aprendizaje automático que se podrían incorporar en el escenario.

7. BIBLIOGRAFÍA

[1] Cisco, «Cisco Annual Internet Report (2018–2023) White Paper,» *Cisco*, 2020.

[2] J. Oltsik, «The Rise of Direct Internet Access (DIA),» CISCO, May 2019. [En línea]. Available: https://security.umbrella.com/esg-report-rise-of-dia/. [Último acceso: 6 June 2023].

[3] OECD, «TELEWORKING IN THE COVID-19 PANDEMIC: TRENDS AND PROSPECTS© OECD 2021,» *OECD*, pp. 1-3, 2021.

[4] Z. Zhang, «Enterprise Networking with Secure Acess Service Edge,» *The Twelfth International Conference on Advances in Future Internet*, 2020.

[5] L. Colombus, «Benchmarking your cybersecurity budget in 2023,» venturebeat, 16 February 2023. [En línea]. Available: https://venturebeat.com/security/benchmarking-your-cybersecurity-budget-in-2023/. [Último acceso: 23 June 2023].

[6] E. Rosen, Cisco-Systems, A. Viswanathan, Force10-Networks, R. Callon y Juniper-Networks, «IETF,» Junuary 2001. [En línea]. Available: https://www.rfc-editor.org/rfc/rfc3031#page-3. [Último acceso: 9 June 2023].

[7] GITST-ETSIT-UPM-Redes-Corporativas, MPLS Construcción y distribución de etiquetas, Madrid: Moodle.

[8] M. Bhandure, G. Deshmukh, P. Varshapriya y IJERA, «Comparative Analysis of Mpls and Non -Mpls Network,» August 2013. [En línea]. Available: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=02f165860b145c5d68628b1d4f5da7646472d2b5. [Último acceso: 9 June 2023].

[9] R.Perez, A.Zabala y A.Banchs, «Alviu: An Intent-Based SD-WAN Orchestrator of Network Slices for Enterprise Networks,» *2021 IEEE 7th International Conference on Network Softwarization (NetSoft)*, pp. 211-215, 2021.

[10] J. Manar, S. Taranpreet, S. Abdallah, A. Rasool y L. Yiming, «Software defined networking: State of the art and research challenges,» *Computer Networks*, vol. 72, nº 1389-1286, pp. 74-98, 2014.

[11] Aryaka, «Migrating from MPLS to SD-WAN,» *Aryaka Whitepaper*.

[12] R. Naggi y F. Sales, Journey Into the World of SASE, Palo Alto: VMware Press, 2021.

[13] P.Ranaweera, A.D.Jurcut y M.Liyanage, «Survey on Multi-Access Edge Computing Security and Privacy,» *IEEE Communications Surveys & Tutorials*, vol. 23, nº 2, pp. 1078-1124, 2021.

[14] A. Lerner y J. Watts, «SASE, SSE, SDWAN – What the what, now?,» Gartner, 5 April 2022. [En línea]. Available: https://blogs.gartner.com/andrew-lerner/2022/04/05/sase-sse-sdwan-what-the-what-now/. [Último acceso: 20 May 2023].

[15] T. Joos, «SASE and PoPs protect us against next generation cyber attacks,» inCyber, 17 February 2023. [En línea]. Available: https://incyber.org/en/sase-pops-protect-us-against-next-generation-cyber-attacks/. [Último acceso: 5 June 2023].

[16] D. Férnandez, «VNX Language Reference,» May 2013. [En línea]. Available: https://web.dit.upm.es/vnxwiki/index.php/Reference#.3Chost_mapping.3E. [Último acceso: 8 June 2023].

[17] VNX-TEAM y D. Fernandez, «Virtual Networks over linuX (VNX) web site,» UPM, 7 September 2020. [En línea]. Available: http://web.dit.upm.es/vnxwiki/index.php/Main_Page. [Último acceso: 5 June 2023].

[18] Open vSwitch, «Production Quality, Multilayer Open Virtual Switch,» The Linux Fundation, 2016. [En línea]. Available: https://www.openvswitch.org/. [Último acceso: 9 June 2023].

[19] Juniper, «What is VXLAN?,» [En línea]. Available: https://www.juniper.net/us/en/research-topics/what-is-vxlan.html. [Último acceso: 9 June 2023].

[20] Juniper-Networks, «IP FABRIC EVPN-VXLAN REFERENCE ARCHITECTURE,» *JUNIPER*, pp. 3-10, 2019.

[21] Huawei, «Huawei Enterprise,» Huawei, 20 Decemmber 2020. [En línea]. Available: https://support.huawei.com/enterprise/en/doc/EDOC1100086966. [Último acceso: 6 June 2023].

[22] CheckPoint, «¿Qué es un sistema de detección de intrusos (IDS)?,» CheckPoint, [En línea]. Available: https://www.checkpoint.com/es/cyber-hub/what-is-an-intrusion-detection-system-ids/. [Último acceso: 5 June 2023].

[23] K. Sharma, «Intrusion Detection System (IDS): Types, Techniques, and Applications,» knowledgehut, 15 February 2023. [En línea]. Available: https://www.knowledgehut.com/blog/security/intrusion-detection-system. [Último acceso: 5 June 2023].

[24] J. Griffin, «What Is an Intrusion Detection System (IDS)?,» LOGICALREAD, 29 July 2021. [En línea]. Available: https://logicalread.com/intrusion-detection-system/. [Último acceso: 5 June 2023].

[25] SEGU DIT-UPM, Proteger Arquitectura de protección V1.9, Madrid: Moodle, 2023.

[26] SNORT, «SNORT,» Cisco, 2023. [En línea]. Available: https://www.snort.org/. [Último acceso: 5 June 2023].

[27] Cyvatar, «cyvatar,» CYBATAR.AI, 27 January 2022. [En línea]. Available: https://cyvatar.ai/write-configure-snort-rules/. [Último acceso: 5 June 2023].

[28] RYU, «COMPONENT-BASED SOFTWARE DEFINED NETWORKING FRAMEWORK. Build SDN Agile,» Ryu, 2017. [En línea]. Available: https://ryu-sdn.org/. [Último acceso: 6 June 2023].

[29] VirtualBox, «VirtualBox,» Oracle, 2023. [En línea]. Available: https://www.virtualbox.org/. [Último acceso: 6 June 2023].

[30] LXC, «LXC,» Canonical LTD, 2023. [En línea]. Available: https://linuxcontainers.org/lxc/introduction/. [Último acceso: 5 June 2023].

[31] K. Henrik-Lund, «VXLAN Security or Injection,» *xlan-bornhack-2018.tex*, pp. 2-16, 2018.

[32] OPEN NETWORKING FOUNDATION, OpenFlow Switch Specification, 2015; The Open Networking Foundation, 2015.

[33] anarkiwi, «ryu/app,» RYU, 2021. [En línea]. Available: https://github.com/faucetsdn/ryu/tree/master/ryu/app. [Último acceso: 9 June 2023].

ANEXO A: ASPECTOS ÉTICOS, ECONÓMICOS, SOCIALES Y AMBIENTALES

A.1 INTRODUCCIÓN

El sistema diseñado está centrado en el contexto de dos campos, el de las redes y el de la ciberseguridad. Los problemas que se pretender cubrir con este TFT han sido por una parte el de optimizar las redes tradicionales las cuales no son eficientes ante los nuevos escenarios emergentes de la nube, además se pretende mejorar tanto el coste como la dificultad del despliegue y aumentar la seguridad en esta cadena.

A.2 DESCRIPCIÓN DE IMPACTOS RELEVANTES RELACIONADOS CON EL PROYECTO

Los impactos más relevantes de este TFT son por una parte la mejora de la seguridad en la red y flexibilidad del despliegue a red de una nueva corporación y a la vez la simplificación de la misma. En este punto se van a tratar los distintos impactos más relevantes para la sociedad.

- Impacto ético, cabe destacar la privacidad de los datos. El uso de soluciones SASE permite proteger tanto a los usuarios como a las corporaciones de filtraciones o ataques. Además, se recuerda el concepto de ZERO TRUST el cual permite proteger de manera segmentada los recursos de la corporación.

- Aspecto social, al ser una solución que permite acceder a recursos desde cualquier lugar y momento puede permitir una reducción en la escasez de infraestructura en algunas zonas.

- El impacto más importante puede residir en el económico ya que SASE permite reducir de forma considerable los costes operativos de desplegar una red propia para la corporación. Desde un punto de vista de ahorro de recursos de red, se debería estudiar las conclusiones que provocaría el tener una red más eficiente y flexible.

- Impacto medioambiental, se puede justificar como la reutilización de recursos de seguridad permite ahorrar infraestructura de despliegue.

A.3 ANÁLISIS DETALLADO DE ALGUNO DE LOS PRINCIPALES IMPACTOS

La eficiencia de SASE permite reducir los costes operativos de las organizaciones, no solo de manera directa, sino que además la simplificación de la red hace que el trabajo de mantenimiento y organización se simplifique. Se debe recordar que gracias a SASE se reducen los gastos en despliegue de infraestructura ya que estos se virtualizan e implementan en otras ubicaciones.

A.4 CONCLUSIONES

Este TFT se desarrolla en torno a dos pilares principales que son la ciberseguridad y las redes. Estos aspectos son críticos para cualquier empresa que quiera optimizar y proteger sus recursos. Por último, se observa cómo esto podría ahorrar distintos recursos, en concreto en el económico, y proteger a las empresas.

ANEXO B: PRESUPUESTO ECONÓMICO

ANEXO B: PRESUPUESTO ECONÓMICO				
COSTE DE MANO DE OBRA DIRECTO (coste directo)		Horas	Precio/hora	Total
		400	13,50 €	5.400,00 €
COSTE DE RECURSOS MATERIALES (coste directo)	Precio de compra	Uso en meses	Amortización (en años)	Total
Ordenador personal	900,00 €	8	5	120,00 €
COSTE TOTAL DE RECURSOS MATERIALES				120,00 €
GASTOS GENERALES (costes indirectos)	15,00%	sobre CD		828,00 €
BENEFICIO INDUSTRIAL	6,00%	sobre CD+CI		380,88 €
SUBTOTAL PRESUPUESTO				6.728,88 €
IVA APLICABLE			21,00%	1.413,06 €
TOTAL PRESUPUESTO				8.141,94 €

ANEXO C: POSIBLE AMPLIACIÓN DEL ESCENARIO