UNIVERSIDAD DEL VALLE DE GUATEMALA

Facultad de ingeniería

Implementación de un sistema de monitoreo sobre servicios Ethernet e IP

Trabajo de graduación en modalidad de Trabajo Profesional presentado por

Jonathan Alexander Robles Pineda

para optar al grado académico de Licenciado en Ingeniería Electrónica

Guatemala,

2015

Implementación de un sistema de monitoreo sobre servicios Ethernet e IP

UNIVERSIDAD DEL VALLE DE GUATEMALA

Facultad de ingeniería

Implementación de un sistema de monitoreo sobre servicios Ethernet e IP

Trabajo de graduación en modalidad de Trabajo Profesional presentado por

Jonathan Alexander Robles Pineda

para optar al grado académico de Licenciado en Ingeniería Electrónica

Guatemala,

2015

Vo. Bo. :

(f) _____

Carlos Alberto Esquit Hernandez
Asesor

Tribunal Examinador:

(f) _____

Carlos Alberto Esquit Hernandez

(f) _____

José Augusto Sanchez

(f) _____

Juan Carlos Montenegro

Fecha de aprobación: Guatemala, 27 de noviembre de 2015

ÍNDICE

LISTADO DE FIGURAS

LISTADO DE CUADROS

RESUMEN

El objetivo de este trabajo fue el desarrollo e implementación de un sistema de monitoreo para una empresa de telecomunicaciones (ISP) sobre enlaces privados de naturaleza Ethernet e IP dentro de una red MPLS. La base del monitoreo IP fue el servicio RPM, mientras que para el monitoreo Ethernet, fue la implementación de un algoritmo de detección y diagnóstico de direcciones MAC y generación de gráficas de tráfico.

En la actualidad, cualquier organización depende de las telecomunicaciones a través de servicios prestados por medio de enlaces de datos e internet. Sin la garantía de recibir enlaces de alta disponibilidad y funcionales, las organizaciones se alejan del alcance de sus objetivos cualesquiera que estos sean. Los proveedores, por tanto, están obligados a ofrecer servicios de calidad y con una alta disponibilidad. Se justifica este trabajo argumentando que un sistema de monitoreo permite al proveedor cumplir con estos objetivos.

Los resultados mostraron un sistema de monitoreo dividido en dos subsistemas y cuyas herramientas permiten el cumplimiento del objetivo general. El monitoreo basado en RPM es ideal para servicios IP, pero no resulta adecuado para servicios Ethernet a pesar de ser factible. Se concluyó que el sistema de monitoreo una vez implementado contribuye a la reducción de los tiempos de respuesta y de diagnóstico de incidentes, traduciéndose en una mejora en la calidad del soporte y a su vez en una mejora de imagen de la empresa.

I. INTRODUCCIÓN

El monitoreo de redes consiste en supervisar los enlaces de transmisión de datos de manera simple y eficiente, con la finalidad de detectar cualquier eventualidad en la red que perjudique los servicios prestados a través de la misma. Las empresas de telecomunicaciones prestan servicios hoy en día, a través de enlaces de datos de naturalezas distintas, dentro de los que sobresalen, los enlaces Ethernet e IP. El objetivo general de este Trabajo de Graduación fue desarrollar e implementar un sistema de monitoreo que permitiera detectar incidentes sobre servicios que se prestan por medio de enlaces de datos privados Ethernet e IP dentro de una red MPLS, mejorando el análisis y disminuyendo el tiempo de detección y diagnóstico de fallas, con el fin de lograr una mejora en la calidad de soporte e imagen de la empresa.

Se desarrolló el sistema para enlaces del tipo Ethernet e IP, implementados dentro de una red MPLS. Por ello, se utiliza la denominación MPLS IP-VPN para el caso IP, y MPLS VPLS para el caso Ethernet, a lo largo de este trabajo. Debido a la contrastante naturaleza entre enlaces Ethernet e IP, el sistema de monitoreo se desarrolló en dos fases distintas, o subsistemas. El subsistema MPLS IP-VPN se basó en la implementación del servicio RPM (*Real-time Performance Monitoring*) bajo el tipo de prueba *ICMP-ping* El subsistema MPLS VPLS tuvo su base en la generación de gráficas de tráfico y el desarrollo e implementación de un algoritmo de diagnóstico inicial basado en la detección de direcciones MAC.

La teoría desarrollada en este trabajo busca familiarizar al lector con conceptos básicos acerca del protocolo IP y del estándar Ethernet. Se desarrolla el tema de redes de área amplia (WAN), dentro de lo que se destaca su definición, los términos comúnmente utilizados, anchos de banda típicos, tipos de conexiones y de soporte WAN. Resalta en esta última categoría, MPLS, describiendo así enlaces MPLS IP-VPN y MPLS VPLS. El marco teórico continúa en una segunda parte, donde habiendo definido conceptos básicos y las tecnologías sobre las que se desenvuelven los servicios, se procede a describir la teoría detrás de las herramientas de monitoreo utilizadas.

El diseño experimental muestra el procedimiento utilizado para cumplir con los objetivos de cada subsistema de monitoreo. Se ocupó primero en desarrollar el sistema de monitoreo de capa 3, y luego el de capa 2. La sección de resultados muestra por cada parte del diseño experimental, los aspectos logrados. En la sección de análisis y discusión de resultados se menciona nuevamente el objetivo general del Trabajo de Graduación como punto de introducción, enfatizando que del mismo se desprendieron todas las actividades realizadas a lo largo del trabajo y se realiza un análisis de los pasos del diseño experimental y de los resultados obtenidos.

Finalmente, se concluye que el sistema de monitoreo implementado contribuye a la reducción de los tiempos de respuesta y de diagnóstico de incidentes, así como el análisis de los mismos, transmitiendo una mejora en la calidad del soporte e imagen de la empresa. El desarrollo del trabajo propuesto estuvo planificado para realizarse durante los meses de junio a octubre del año 2015.

II. OBJETIVOS

A. GENERAL

Desarrollar e implementar un sistema de monitoreo que permita detectar incidentes sobre servicios que se prestan por medio de enlaces de datos privados Ethernet e IP dentro de una red MPLS, mejorando el análisis y disminuyendo el tiempo de detección y diagnóstico de fallas, con el fin de comprobar una mejora en la calidad de soporte e imagen de la empresa.

B. ESPECÍFICOS

- Hacer uso de un sistema de monitoreo existente para servicios prestados por medio de enlaces privados IP, MPLS IP-VPN, aplicándolo a todos los servicios de esta naturaleza que la empresa presta.

- Desarrollar e implementar un sistema de monitoreo que permita detectar incidentes en servicios prestados por medio de enlaces privados Ethernet, MPLS VPLS, y demostrar su nivel de funcionalidad.

- Desarrollar un registro automatizado en equipos y plataformas de monitoreo, en tiempo real e histórico, que mejore el análisis y anteposición a incidentes presentados sobre servicios privados IP y Ethernet.

- Implementar un estándar de identificación en el monitoreo de servicios cuyo fin es organizar la recolección de información, autodescubrimiento de interfaces, RPMs y MACs dentro de un portal dedicado a los clientes.

- Aplicar el servicio RPM-Ping (Real Time Performance Monitoring) sobre todos los enlaces IP-VPN de punto inicial a punto final en equipos de monitoreo asociados, identificando cada RPM de forma estandarizada.

- Generar gráficas RPM sobre todos los servicios IP-VPN que sean monitoreados en una plataforma especializada con registros en tiempo real e histórico.

- Crear archivos syslog dentro de los equipos de monitoreo asociados que almacenen las pruebas fallidas de forma temporal y almacenarlas en servidores syslog de forma permanente por cada enlace punto-multipunto IP-VPN.

- Realizar una investigación exhaustiva por cada servicio IP-VPN con el fin de obtener las IPs correctas en el punto inicial (Central) y punto final (CPE) para un correcto monitoreo.

- Generar lecturas de tráfico automatizadas en las sub-interfaces asociadas, punto a punto, como parte del desarrollo de monitoreo de enlaces Ethernet-VPLS, con el fin de detectar problemas, tales como saturación en el enlace, falta de utilización por parte del usuario final o un mal funcionamiento del mismo al momento de no detectar tráfico.

- Desarrollar un algoritmo de diagnóstico inicial que permita hacer reconocimiento de direcciones MAC en los puntos inicial y remotos de los servicios Ethernet-VPLS con el fin de determinar inoperatividad de los mismos al no detectar MACs en uno o varios puntos.

- Llevar un registro automatizado de los equipos conectados en los puntos inicial y remotos por medio del reconocimiento de direcciones MAC, con el fin de detectar cambios de equipos o de cantidad de direcciones MAC por cada punto que pueda ser indicador de un mal funcionamiento en los servicios Ethernet-VPLS.

- Ofrecer una vía alterna de monitoreo Ethernet-VPLS, comparar y analizar los métodos, argumentado el uso de uno sobre el otro.

- Generar un manual de monitoreo estándar que permita a los ingenieros de soporte implementar el monitoreo e identificación de servicios IP-VPN y Ethernet-VPLS de forma específica y ordenada.

III. JUSTIFICACIÓN

En la actualidad, cualquier empresa u organización, en general, depende de las telecomunicaciones. Las mismas requieren de servicios dados (como enlaces de datos y de internet) por empresas dedicadas a esta área para tener acceso a recursos propios y terceros, con el fin de funcionar como organización, dado que la tendencia actual les ha marcado el rumbo a migrar a un estilo de vida digital. Sin una alta disponibilidad en sus servicios, las mismas pierden económicamente y sus objetivos cualesquiera se dejan de cumplir. En consecuencia, exigen una disponibilidad total sobre los servicios que contratan a empresas de telecomunicaciones. Sin embargo, dichos enlaces están propensos a eventualidades que pueden afectar los servicios. La correcta operación de estos enlaces se hace cada vez más crítica para el éxito de las empresas de telecomunicaciones, a medida que las redes se vuelven más grandes y complejas. Este crecimiento está dado en gran parte al desarrollo tecnológico en el que se vive actualmente.

La empresa para la cual se realizó la implementación de monitoreo presta en la actualidad enlaces de datos privados Ethernet e IP dentro de una red MPLS, a través de diferentes tecnologías y protocolos, como VPLS y BGP respectivamente. Actualmente en la empresa, la forma común a través de la cual se detectan incidentes en estos servicios es por medio de reportes de fallas de los usuarios. Otra forma de detección consiste en la revisión constante de los servicios en los equipos de transmisión (Routers y Switches) asociados. Los ingenieros de soporte de este departamento atienden periódicamente eventualidades por medio de los reportes mencionados. Revisar constantemente los enlaces de los servicios no es una opción viable, debido a que normalmente se consume el tiempo atendiendo fallas activas.

Se implementó un sistema de monitoreo con la finalidad de analizar los incidentes con mayor eficacia, disminuyendo el tiempo de alerta y de diagnóstico de los mismos. No se buscó realizar un monitoreo completo, sino recolectar automáticamente la información necesaria que le pueda indicar al operador de una forma rápida y comprensible si un servicio está operativo o no, y el tipo de atención que le merece. De esta forma, transmitir interés y proactividad a los suscriptores fue un fin importante de este trabajo, evitando un monitoreo constante por parte de los operadores y mejorando sus tiempos de respuesta. Por lo tanto, se buscó comprobar que dicha implementación mejorará la calidad del soporte, dando una mejor imagen de la empresa.

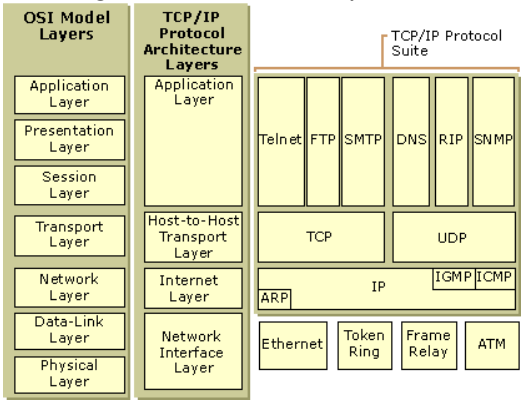
IV. MARCO TEÓRICO

En este apartado, previamente a tratar temas y aspectos teóricos relacionados al desarrollo de monitoreo de enlaces basados en IP o Ethernet, se considera necesario explicar conceptos básicos que se manejan en el campo de las telecomunicaciones, con el fin de familiarizar al lector y proveer un mayor entendimiento de este trabajo.

Estos conceptos abarcan desde la definición de los protocolos IP y Ethernet, pasando por la definición de una WAN y el método MPLS, hasta la definición de herramientas y técnicas de monitoreo como lo son: RPM (Real-time Performance Monitoring), Syslog (registros de eventos), PyEZ (módulo de Python para manejo y automatización de dispositivos Juniper), Netconf (Protocolo utilizado por el sistema operativo de equipos Juniper), Cacti (plataforma de gráficos), y SNMP (Simple Network Manager Protocol).

En la siguiente figura, se especifican las capas del Modelo OSI y del modelo TCP/IP, así como los protocolos asociados a cada una. Para fines de este trabajo, interesa detallar en las capas 2 y 3 definidas dentro del Modelo OSI, así como de protocolos específicos de capa de aplicación (Netconf y SNMP).

Figura#1: Modelos OSI y TCP/IP



A. PROTOCOLO IP

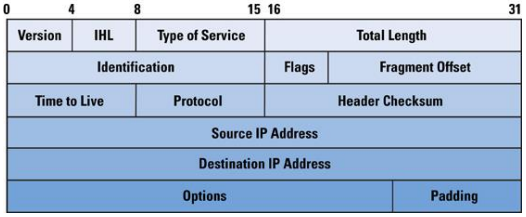
IP (Internet Protocol) es el protocolo de Internet definido por el Departamento de Defensa de los Estados Unidos de América (por sus siglas en inglés DoD), específicamente por la DARPA (Defense Advanced Research Project Agency, agencia del DoD dedicada al desarrollo de nuevas tecnologías) dentro del RFC 791 en septiembre de 1981 y en reemplazo de una versión anterior, RFC 760, en enero de 1980. Es el protocolo estándar principal empleado en las telecomunicaciones y demás redes de datos, definido dentro del modelo DoD, conocido más propiamente como el modelo TCP/IP dentro de la capa de Internet, equivalente a la capa de red (capa 3) del Modelo ISO/OSI.

Este protocolo define las series de reglas que permiten el ruteo entre redes, es decir, la transmisión de paquetes o datagramas a través de distintos segmentos de red, lo que esencialmente permite el internet o conexión entre múltiples redes. La función básica del protocolo IP se resume en entregar un paquete desde una fuente IP, basado únicamente en el destino IP. El protocolo IP

traza el camino que debe seguir un datagrama desde la fuente hasta el destino basado en dos conceptos básicos: el direccionamiento IP y el enrutamiento o ruteo.

Es un protocolo no basado en conexión (connectionless), lo que indica que no utiliza mecanismos para asegurar la entrega de paquetes. Esto debido a que protocolos definidos dentro del Modelo OSI como de capa 4 (TCP) y capa 2 (Ethernet), se encargan de la entrega confiable de segmentos y fragmentos de datos mediante técnicas específicas acorde a la capa en la que se definen.

Figura#2: Encabezado IP



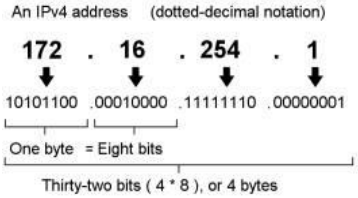
1. DIRECCIONAMIENTO IP. Una dirección IP es un identificador lógico y único para una interfaz de un dispositivo de red (host, switch o router) que forme parte de una red de datos específica. Asignar una IP específica cumple dos funciones: Dar un nombre e identificar la ubicación del dispositivo. Al ser el nombre único, es posile establecer una o varias rutas (a través de protocolos de enrutamiento) que indiquen el/los camino(s) a seguir para ubicar y llegar al elemento de red. Una dirección IP, por tanto, forma parte del camino que un paquete debe seguir para partir de una fuente y llegar a un destino. Los protocolos de enrutamiento definen qué camino(s) tomar, dependiendo de parámetros específicos de cada protocolo, lo cual se define más adelante.

Los diseñadores del protocolo IP definieron un sistema de direccionamiento de 32 bits aún en uso hoy en día y conocido como IP Version 4, IPV4, cuya definición se incluye en el RFC 791. El constante crecimiento del uso público de IPs a través del Internet hizo necesario que se desarrollará un nuevo sistema de direccionamiento conocido como IP Version 6, IPV6, de 128 bits y estandarizado a través del RFC 2460 en 1998. Su uso inició a mediados de la primer década del presente siglo, aunque aún no ha surgido la necesidad de emplearlo a nivel público o de Internet. Cabe destacar que la IANA (Internet Assigned Numbers Authority) asigna espacios específicos de IPV4 globalmente para distintos propósitos (asignación pública y privada, por ejemplo) y designa 5 registros de Internet regionales (RIRs, Regional Internet Registries) para repartir bloques públicos de direcciones IP a Proveedores de servicio de Internet (ISP, Internet Service Provider).

Para fines de este trabajo, interesa definir únicamente el sistema IPV4. Una dirección IPV4 separa los 32 bits que la componen en cuatro octetos. De esta forma, una IP asignada bajo este sistema de direccionamiento se puede leer como un conjunto de 4 números decimales con capacidad de direccionamiento de 2 a la octava potencia, es decir, capacidad de asignación de 0 a 255 por octeto. Por ejemplo, la dirección 192.168.1.1, es una IP en versión 4, definida dentro del RFC 1918 como de uso privado.

La siguiente figura muestra la relación existente entre la lectura decimal y binaria de una IP en versión 4.

Figura#3: Dirección IP en versión 4.



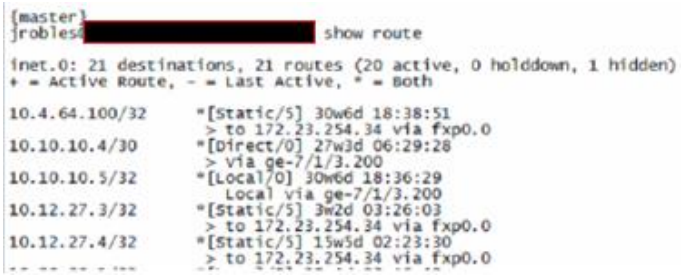
2. ENRUTAMIENTO. El término “ruteo” o “enrutamiento” se refiere a llevar un paquete de un punto a otro a través de distintos segmentos de red dentro de un sistema de telecomunicaciones como lo puede ser una red WAN (Wide Area Network) o una red local (LAN, Local Area Network). El ruteo se basa en dos aspectos esenciales: la dirección IP y el protocolo de enrutamiento. Un conjunto de direcciones IP entre distintos segmentos de red para llegar de un punto a otro, se definen como el camino que el paquete atraviesa y los protocolos de enrutamiento son las señales que indican qué camino(s) se deben seguir para lograr la comunicación. Los dispositivos típicos de ruteo dentro de los elementos de red son los routers. Los routers fueron diseñados para encontrar rutas y definir el mejor camino para llegar a ellas a través de direcciones lógicas (direcciones IP). Dentro de una “Inter-red”, un router es un dispositivo esencial para comunicarse con las distintas redes que forman parte de la misma. A continuación, se detalla un listado de requerimientos mínimos que un equipo de ruteo necesita para cumplir con su propósito:

- Dirección IP destino.
- Routers vecinos de los cuales puede aprender rutas (a través de protocolos de ruteo).
- Rutas posibles a todos los puntos remotos.
- La mejor ruta hacia cada punto remoto (depende del protocolo de ruteo utilizado).
- Cómo mantener, verificar y actualizar la información de ruteo (también depende del protocolo de ruteo utilizado).

Al aprender las rutas, los equipos de ruteo las almacenan dentro de su tabla de rutas, que es básicamente un mapa de cómo alcanzar cada punto remoto o red, del conjunto total de redes del cual forma parte. Una tabla de rutas contiene varios elementos básicos: prefijo de la red destino, próximo salto (“next-hop”) es decir, el router vecino al cual enviar el paquete, la fuente de información de ruteo (protocolo) por medio del cual se aprende la ruta, la interfaz de salida del equipo, la preferencia de ruta o distancia administrativa (número jerárquico que determina qué protocolo de ruteo se debe utilizar sobre otro) y típicamente el tiempo que tiene el equipo de conocer la ruta.

En la siguiente figura se muestra el detalle de una tabla de rutas aprendida por un equipo Juniper, en donde se puede apreciar que muestra los elementos mínimos mencionados:

Figura#4: Tabla de rutas de un equipo Juniper



Si el equipo posee una red capa 3 directamente conectada, el prefijo de dicha red se asigna automáticamente a la tabla de rutas del mismo, puesto que ya la conoce. Si la red es remota, entonces existen dos formas de determinar el camino hacia la misma: a través un ruteo estático o uno dinámico.

a. Ruteo estático. Es el proceso por medio del cual se aprenden rutas al asignarlas manualmente en cada equipo de red capa 3. Su preferencia de ruta o distancia administrativa es por defecto menor a la de protocolos de ruteo dinámico (lo que quiere decir que es preferente sobre estos) pero mayor a las de rutas directamente conectadas.

A continuación, se presentan las ventajas del uso de un ruteo estático:

- Bajo consumo del procesador (CPU) del elemento de red: el ruteo estático es permanente, por lo que no se realiza ningún tipo de mecanismos o cálculos de rutas óptimas para actualización o determinación de la mejor ruta.
- No hay consumo del ancho de banda en el enlace: Por la misma razón, los routers no hacen uso del ancho de banda, lo cual puede resultar bastante atractivo en un enlace WAN que provee un ISP, por ejemplo.
- Añade seguridad en el sentido que el administrador puede determinar a través del ruteo, qué redes puede alcanzar el equipo y vice-versa.

A continuación, se detallan las desventajas del ruteo estático:

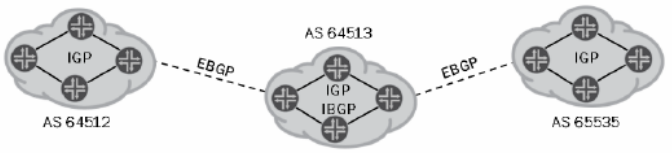
- El administrador de la red debe conocer exhaustivamente todo el mapa que la compone.
- Al añadir una red al conjunto de redes, esta se debe especificar manualmente en cada equipo que compone la red entera.
- En consecuencia al segundo punto, no resulta apropiado manejar un ruteo estático dentro de redes grandes.

b. Ruteo dinámico. Es el mecanismo utilizado por protocolos de ruteo para colocar y actualizar la tabla de rutas de un dispositivo de red capa 3 de forma automatizada. Como no puede ser de otra manera, las ventajas y desventajas de un ruteo dinámico son las de un ruteo estático pero invertidas. Un protocolo de ruteo establece reglas específicas para determinar la mejor ruta para cada destino y para actualizarlas.

Existen dos tipos de protocolos de ruteo dínamicos: EGPs (Exterior Gateway Protocols) e IGPs (Interior Gateway Protocols). Protocolos de tipo IGP, son utilizados para intercambiar paquetes dentro de dispositivos de capa de red ubicados dentro del mismo sistema autónomo (AS), mientras que protocolos de tipo EGP son utilizados para interconectar redes entre distintos sistemas autónomos o infraestructuras de red independientes.

La figura a continuación, detalla un ejemplo de la como ocurre la comunicación entre equipos dentro y fuera de un sistema autónomo:

Figura#5: Comunicación entre y dentro de sistemas autónomos (AS)



A continuación, se resume un listado de protocolos clasificados por EGP e IGP:

IGPs:

- RIP (Routing Information Protocol): Protocolo de métrica vector-distancia, es decir, basado en los saltos o cantidad de routers que debe atravesar, para escoger la ruta óptima.
- OSPF (Open Shortest Path First): Protocolo de métrica estado-enlace (link-state), que se basa en el ancho de banda de los enlaces para escoger la ruta óptima.
- IS-IS (Intermediate System to Intermediate System): Protocolo en el que los paquetes no son encapsulados a nivel IP, sino directamente a nivel de capa de enlace de datos (capa 2).

EGPs:

- BGP (Border Gateway Protocol): Protocolo de borde para la interconexión de distintos sistemas autónomos. Es clasificado como un protocolo ruta-distancia y vector-distancia. Este protocolo basa sus decisiones en rutas, políticas de ruteo, o reglas configuradas por el administrador de la red. Este protocolo se detalla en profundidad más adelante.

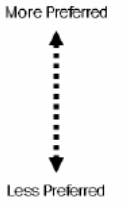
Dentro de un elemento de capa de red, pueden existir múltiples rutas para un mismo destino o prefijo. Una de las formas más comunes en los que un elemento escoge la ruta activa es por medio de la distancia administrativa o preferencia de ruta. Así por ejemplo, si conoce una ruta por ruteo estático y por RIP, el elemento de red escoge por defecto la ruta estática como la ruta activa. La preferencia es configurable entre dispositivos o routers.

La figura a continuación, detallan la preferencia de ruta en equipos Juniper:

Figura#6: Preferencia de ruta (equipos Juniper)

Route Preference Values

Routing Information Source	Default Preference
Direct	0
Local	0
Static	5
OSPF internal	10
RIP	100
OSPF AS external	150
BGP (both EBGP and IBGP)	170



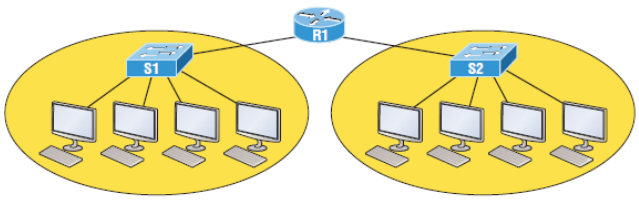
B. ETHERNET

Ethernet es un estándar y una familia de protocolos y tecnologías definido dentro de la capa de enlace de datos (data link) y capa física (capa 1) del Modelo OSI. Es utilizado ampliamente para permitir la conexión entre dispositivos a nivel de redes de área local (LAN) y redes de área metropolitana (redes Metro o MAN) y con cierta aplicación en redes de área amplia (WAN). Fue introducido en 1980 y estandarizado como IEEE 802.3. A diferencia del Protocolo de internet (IP) que define las direcciones IP como únicas dentro de cada conjunto de redes (al menos a nivel de direccionamiento privado), Ethernet define las direcciones de control de acceso a media (MAC, Media Access Control) como identificadores únicos de las interfaces (NIC, Network Interface Cards) de los elementos de red (Routers, Switches, Hosts) a nivel global. Es decir que la interfaz de un dispositivo jamás podría repetirse en ningún otro dispositivo en el mundo. Para ello, Ethernet define un estándar de asignación.

Ethernet permite la conectividad entre dispositivos dentro de un mismo segmento de red o dominio de Broadcast con el uso exclusivo de direcciones MAC como método de direccionamiento, dejando a un lado el uso de direcciones IP. A esto se le conoce como direccionamiento Ethernet o de capa 2. Si en capa 3, el Router es el elemento de red principal, en capa 2 el Switch o Conmutador es el dispositivo encargado del direccionamiento MAC. Dentro de los protocolos definidos por Ethernet destaca CSMA/CD (Carrier Sense Multiple Access with Colission Detection) que define el algoritmo a implementar al momento de ocurrir colisiones en un medio compartido. Esto es común entre interfaces configuradas en modo Simplex o Half-Duplex. Cabe destacar que la configuración a Full-Duplex elimina las colisiones y por tanto, el uso de este protocolo.

En la siguiente figura se puede apreciar la diferencia de impacto entre un Router y un Switch. Un Router separa Broadcast Domains y por tanto hace necesario el ruteo entre una red y otra por medio de direccionamiento IP. El Switch en cambio separa Collision Domains dentro de un Broacast Domain único, permitiendo la comunicación entre dispositivos a nivel de direccionamiento MAC. Teniendo claros los conceptos de Dominio de Colisión y Dominio de Difusión, pueden observarse 10 y 2, respectivamente en la figura.

Figura#7: Impacto de Routers y Switches en una red



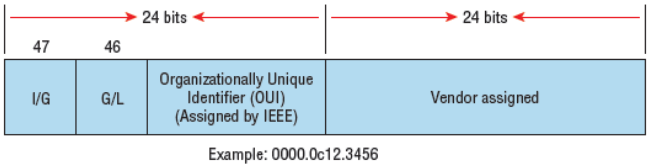
Manejando el concepto estrictamente definido a nivel de capa 2, un dominio de difusión es equivalente a una red de área local, existiendo dos LAN en la Figura#7. Sin embargo el término LAN es frecuentemente utilizado para definir un conjunto de redes dentro de una misma infraestructura física, hogar u oficina, es decir, cuya distancia física no es considerable.

Por otro lado, se ha mencionado que Ethernet define protocolos a nivel de capa 2 y capa 1. Sin embargo, para fines de este trabajo, interesa únicamente conocer Ethernet a nivel de capa de Enlace de Datos (capa 2), específicamente dos conceptos básicos como lo son el direccionamiento Ethernet y la conmutación a nivel de capa 2 (Layer 2 Switching).

1. DIRECCIONAMIENTO ETHERNET. Una de las funciones principales que se buscan dentro del estándar Ethernet a nivel de capa 2 es el direccionamiento MAC o de hardware. Tal y como se mencionó, una dirección MAC es única a nivel mundial y propia de cada interfaz física de cada elemento de red (hardware). Una dirección de Control de Acceso a Media (MAC) está compuesta por 48 bits y es por defecto representada de forma hexadecimal.

En la siguiente Figura, se puede apreciar la forma de asignación de una MAC. El campo OUI es asignado por la IEEE a una organización específica dedicada a la fabricación de elementos de red. Los bits 47 y 46 resultan en una excepción a la regla que afirma que una MAC es única, aunque esto suele ser así. El bit I/G (Individual/Group) al ser 0 indica que la MAC es única de una NIC de un elemento de red. AL ser 1, indica que la MAC puede ser Mulitcast o Broadcast, es decir que hace referencia a un conjunto de NICs dentro de un dominio de difusión. El bit G/L (Global/Individual) al ser 0 representa una MAC administrada globalmente, es decir, asignada por la IEEE y al ser 1, representa una MAC asignada por una entidad individual y para uso exclusivo de dicha entidad.

Figura#8: Asignación de dirección MAC

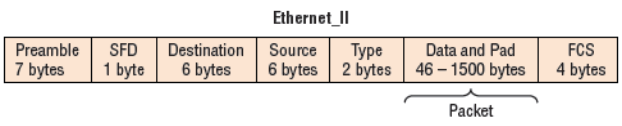


En resumen, una MAC puede ser única o no, pero una NIC o interfaz física dentro de un elemento de red siempre tendrá una MAC única.

2. CONMUTACIÓN CAPA 2 (LAYER 2 SWITCHING). Si en capa 3, a la distribución de paquetes entre elementos de red se le conoce como ruteo, en capa 2, a la distribución de tramas entre equipos (switches) se le denomina conmutación capa 2 (Layer 2 Switching). Un switch de capa 2 realiza esta tarea con mayor rapidez que un router, debido a que no revisa la información del encabezado de capa 3 y por tanto tampoco absorbe tiempo en determinar el siguiente salto (next hop) para escoger la ruta óptima (en el caso de ruteo dinámico). En lugar de esto, únicamente revisa la dirección física de destino de la trama previo a determinar enviarla a un puerto específico del equipo capa 2 (forwarding), a todos los puertos del equipo excepto por el cual salió (flooding) o descartar la trama (discard). Es importante mencionar que los switches proveen un dominio de colisión dedicado para cada uno de sus puertos o interfaces físicas y así mismo, un ancho de banda exclusivo para cada uno.

En la siguiente figura se muestra una trama de datos (Ethernet II) típica. Para fines de este trabajo, interesa prestar atención solamente a los campos de direcciones físicas fuente y destino, y el hecho que la trama contiene el paquete de datos evaluado en el proceso de ruteo (encapsulación).

Figura#9: Encabezado Ethernet II



La distribución de tramas a nivel de capa 2 se compone de tres funciones básicas:

a. Aprendizaje de MACs (Adress Learning). Una tabla MAC es el equivalente en capa 2 a una tabla de rutas en capa de red. A diferencia de una tabla de rutas, una tabla MAC únicamente se compone de tres elementos: la dirección MAC, el puerto lógico o físico del cual la aprende y la VLAN (Virtual LAN) asociada. Los switches capa 2 almacenan la dirección MAC fuente en su tabla MAC al momento de evaluar el envío de una trama recibida a través de uno de sus puertos.

En la siguiente figura, se muestra una Tabla MAC típica de un Switch Juniper. Como se puede apreciar, además de los parámetros mencionados, se incluyen dos adicionales: el modo de aprendizaje (Type) que puede ser estático (configurada manualmente), aprendida por envío a todos los puertos (Flood, se verá a continuación) o aprendida dinámicamente (Learned) por el proceso que se detalla en este aparatado; y el tiempo de aprendizaje de la MAC (Age).

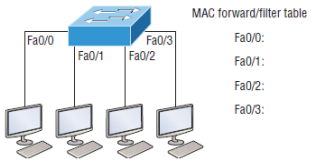
Figura#10: Tabla MAC en equipo Juniper

```
{master:0}
jrobles@jun[redacted]> show ethernet-switching table | match Probe-SLA
  Probe-SLA          *                 Flood             - All-members
  Probe-SLA          00:15:2b:2b:11:6a Learn             0 ge-0/0/24.0
  Probe-SLA          00:17:cb:cc:b8:46 Learn             0 ge-0/0/47.0
  Probe-SLA          00:d0:bb:d0:9a:e0 Learn             0 ge-0/0/9.0
```

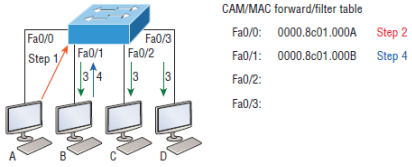
En un inicio, la tabla MAC del equipo de red capa 2 se encuentra vacía. Cuando el switch recibe una trama a través de uno de sus puertos almacena la MAC fuente en su tabla MAC, pero al no tener registrada aún la MAC destino, envía la trama a través de todos sus puertos, exceptuando el que la recibió. El host o dispositivo que posee la MAC responde a través del puerto del switch al que está conectado, enviando una trama. El switch ahora almacena esa MAC (ya que en esta trama, es la fuente) en su tabla de MAC, y al ya haber almacenado la que ahora corresponde al destino, únicamente envía la trama a través de ese puerto. Así, se forma una conexión directa entre esos dos puertos. Este proceso se repite cada vez que resulte necesario establecer dichas conexiones entre los puertos a los que un host se conecte.

Las siguientes figuras muestran el proceso mencionado y que se detalla en los siguientes pasos a continuación:

Figura#11: Estado inicial de una tabla MAC en un Switch



Figura#12: Proceso de aprendizaje de MACs



- El host A envía una trama a través del puerto de conexión al switch.

- El switch evalúa la MAC fuente y la almacena en su tabla MAC (en este ejemplo, la asocia al puerto Fa0/0).
- El switch, al no conocer la MAC destino, envía la trama a través de todos sus puertos excepto del cual la recibió.
- El host destino que posee la MAC responde enviando una trama de vuelta a través del puerto respectivo. El switch al recibir la trama, almacena la MAC (pues es ahora reconocida como MAC fuente) en la tabla MAC, relacionándola al puerto correspodiente (en este caso, Fa0/1).

b. Decisiones de envío/filtrado (Forward/Filter). Cuando una trama es recibida a través de uno de los puertos de un switch, la MAC destino se compara con la tabla MAC. Si la MAC coincide entonces el switch envía la trama únicamente a través del puerto correspondiente, indicado por la tabla. Esto evita el envío a través de todos los puertos, ahorrando ancho de banda. A este proceso se le conoce como filtrado de tramas. En cambio, si la tabla MAC no logra asociar la MAC destino, la trama es entonces enviada al resto de puertos. Si se recibe una respuesta, la MAC es almacenada. Es importante recalcar que durante este proceso, la MAC fuente es siempre evaluada para asociarse a la tabla MAC de no tenerla en el registro.

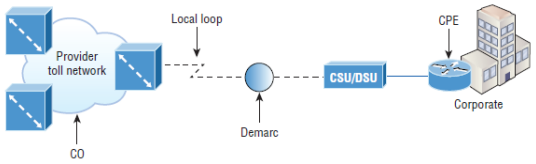
c. Prevención de bucles (Loop Avoidance). Cuando múltiples conexiones son creadas entre switches por propósitos de redundancia, bucles no deseados pueden ocurrir. El protocolo Spanning Tree es utilizado para evitar los bucles mientras permite la redundancia. No se detalla a profundidad en este apartado dado que no corresponde dentro de los objetivos de este trabajo.

C. REDES WAN

Una red de área amplia (Wide Area Network) permite la conexión remota entre dos o más redes de área local. Una característica de una WAN es la distancia física entre sus puntos de conexión. Por otro lado, también está el ancho de banda que se maneja a través de una WAN es considerablemente mayor. Sin embargo, una característica distintiva es el hecho que mientras en una LAN, por ejemplo, se es dueño de la infraestructura de la misma; para establecer una conexión WAN es necesario utilizar la de un Proveedor de Servicio de Internet (ISP).

La imagen a continuación muestra un diagrama en el que se detalla un ejemplo de una WAN y cuyos términos se explican en el próximo apartado.

Figura#13: Elementos de una red WAN



1. TÉRMINOS WAN. A continuación se detalla términos básicos utilizados a nivel de redes WAN:

- Equipo en Sitio de Cliente (CPE, Customer Premises Equipment): Es el equipo que forma parte final de la WAN dentro de las instalaciones de cliente. Puede ser propiedad del mismo (caso en el cual, el ISP no lo gestiona) o del proveedor del servicio.

- Punto de demarcación (Demarc): Es el punto a partir del cual finaliza la responsabilidad del proveedor del servicio e inicia la del cliente o suscriptor. Dependiendo de cuál de las partes lo gestiona y es propietario del CPE, el punto de demarcación puede ser el mismo CPE o bien en el convertidor de medios (CSU/DSU), también instalado en el sitio del cliente.

- Última Milla (Last Mile): Definida dentro de la Figura#4.13 como Local Loop, es la conexión entre el punto de demarcación y el equipo o nodo más cercano al proveedor del servicio, conocido como Equipo Borde (PE, Provider Edge).

- Equipo Borde de Proveedor (PE): Es el equipo que interconecta la red del proveedor con la del suscriptor a través de la Última Milla. Usualmente es referido como Oficina Central (Central Office) o Punto de Presencia (POP, Point of Presence).

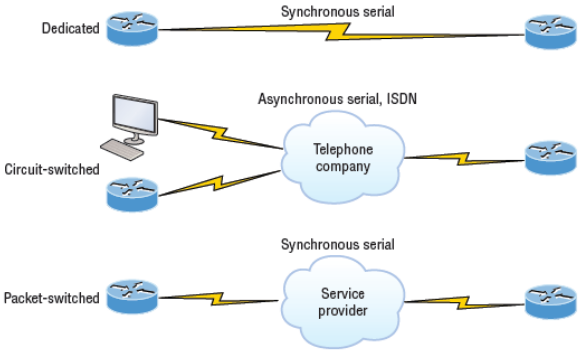
- Red Núcleo (Core Network o Toll Network): Es la colección de elementos de red (Routers, Switches y Líneas Troncales) que forman parte de la red del proveedor y a través de la cual presta todos su servicios.

2. ANCHO DE BANDA EN CONEXIONES WAN. A nivel de WAN en el ámbito de las telecomunicaciones, es de vital importancia estar familiarizado con términos referentes al ancho de banda manejado, tal y como se detalla a continuación.

- DS0 (Digital Signal 0): Señal digital 0 es la tasa básica de 64 Kbps equivalente a un canal o línea de voz/datos, siendo la capacidad mínima transmitida dentro de un circuito digital. El término entra dentro del sistema T-Carrier. El equivalente en E-Carrier es E0.
- T1: Referido como DS1, es una capacidad que multiplexa 24 DS0 para formar una capacidad de 1.54Mbps.
- E1: Equivalente a T1 dentro del sistema E-Carrier, esta capacidad de 2.048Mbps multiplexa 32 canales DS0.
- T3: Conocido como DS3, es una capacidad de 44.736Mbps al reunir 28 DS1 o 672 DS0.
- STM-1/OC-3: Capacidad que reúne 3 DS3, equivalente a 2016 DS0 para un ancho de banda de 155.52Mbps.
- STM-4/OC-12: Capacidad de 622.08 Mbps al reunir 4 STM-1 y equivalente a 8064 DS0.
- STM-16/OC-48: Capacidad de 2.5 Gbps al reunir 4 STM-4 y equivalente a 32,256 DS0.
- STM-64/OC-192: Capacidad de 10 Gbps al reunir 4 STM-16.

3. TIPOS DE CONEXIONES WAN. Existen tres tipos de conexiones WAN, de las cuales parten el resto de tecnologías, soporte y métodos de encapsulación utilizados para la transmisión de datos a nivel de redes de área amplia. La figura a continuación ilustra los tipos de conexiones, seguida de una breve descripción de cada una.

Figura#14: Tipos de conexiones WAN



- Línea Contratada (Leased Line): Es una conexión permanente y dedicada dada por un ISP y típicamente utilizada para interconectar dos sedes remotas. Una Línea Contratada está siempre activa y dado que la conexión no se utiliza para la comunicación de ningún otro suscriptor, el proveedor del servicio puede asegurar la calidad del servicio. Un ejemplo típico de una Línea Contratada es un canal T1.

- Circuit Switching: Conexión dedicada que se establece durante la transmisión de datos a través de la red Core del proveedor del servicio. El ejemplo más claro de una red de este tipo es el sistema telefónico, el cual unifica segmentos de enlaces para establecer una línea durante cada llamada. Este tipo de circuitos es ideal para comunicaciones en tiempo real, en donde la transmisión debe ser rápida y llegar en un orden específico.

- Packet Switching: Conexión dentro de la cual los mensajes son divididos en paquetes conforme son enviados. Cada paquete es transmitido individualmente a través de la red Core del proveedor del servicio e incluso tomando distintas rutas hasta llegar a su destino. Una vez llegan, los paquetes son reconstruidos al mensaje original ya que no se garantiza que lleguen en el mismo orden. Este tipo de conexión es ideal para aplicaciones que soporten cierto retraso en la transmisión, como es el caso de páginas web o correo electrónico, ya que es efectivo y robusto, en el sentido que utiliza protocolos que aseguran el envío completo y correcto de la información. A pesar que el protocolo IP no es orientado a conexión como tal, las redes Packet Switching utilizan el protocolo TCP, que convierte a las redes IP de este tipo en orientadas a conexión.

4. SOPORTE WAN. En la actualidad, existen distintas tecnologías, protocolos, métodos y formas de encapsulamiento para la transmisión de información a nivel de redes WAN. Dependiendo del tipo de interfaz en los equipos de red, pueden existir distintos métodos y tecnologías para encapsular los paquetes. Por ejemplo, en interfaces seriales es típico encontrar protocolos como HDLC, PPP y Frame Relay, aunque en otro tipo de interfaces se puede utilizar otros como ATM, MPLS y Metro Ethernet. Últimamente, se observan cada vez menos conexiones seriales en redes WAN debido a que resultan menos efectivas para el ISP en comparación a interfaces Fast Ethernet o Giga Ethernet. A continuación, se detalla brevemente los protocolos y métodos principales para soporte a nivel de área amplia.

- Frame Relay: Tecnología del tipo Packet-Switching de inicio de la década de 1990. Puede correr a velocidades desde 64Kbps hasta 45Mbps. Provee características para asignación dinámica de ancho de banda y congestión de control.

- HDLC (High-Level Data Link Control): Tecnología cuyo encabezado no tiene identificación del tipo de protocolo cargado dentro de la encapsulación. Debido a ello, cada vendedor o fabricante del equipo desarrolla su propia forma de identificar el protocolo de capa de red, siendo así cada uno dueño de su propio protocolo HDLC.

- PPP (Point to Point Protocol): Protocolo utilizado para conexiones punto a punto entre equipos de diferentes fabricantes, dado que las versiones HDLC son propietarias de cada uno. Utiliza un campo dentro de su encabezado para identificar el protocolo de capa 3.

- ATM (Asynchronous Transfer Mode): Método creado para aplicaciones susceptibles al retardo, proveyendo transmisiones simultáneas de voz, datos y video. Típicamente si se correo Frame Relay, esto se hace sobre ATM.

- MPLS (Multi Protocol Label Switching): Mecanismo que emula propiedades de Circuit Switching y Packet Switching. Es un mecanismo de conmutación (switching mechanism) que impone etiquetas (labels) a los paquetes a transmitir y estos son utilizados para envío. Las etiquetas son asignadas en los equipos borde del proveedor del servicio (PE) y transmitidos dentro de su red MPLS basado únicamente en dichas etiquetas. Las etiquetas corresponden a un camino hacia el destino, que es usualmente una dirección IP. MPLS fue diseñado para ser compatible independientemente del protocolo de capa 3 utilizado. En redes grandes, el beneficio de asignar las etiquetas únicamente en los equipos borde es que sólo estos realizan acciones de ruteo, ahorrándole la tarea a los equipos Core, cuya función puede verse como la de un equipo capa 2 (switch). El resultado es una transmisión rápida de paquetes a través de la red del ISP, razón por la cual muchas compañías han reemplazado las conexiones de Frame Relay y ATM por redes MPLS. Por último MPLS ofrece conectar una WAN por Ethernet, a lo cual se le conoce como Ethernet sobre MPLS, EoMPLS.

- Metro Ethernet: Red de área metropolitana basada en estándares Ethernet y que por tanto, entrega el servicio a nivel de capa 2. Metro Ethernet basado en MPLS es usado por un ISP para interconectar dos sedes a nivel metropolitano por medio cable Ethernet o de fibra óptica. Parte de una conexión Ethernet en el sitio del cliente, pasa a MPLS a través del proveedor y se entrega nuevamente por Ethernet en el punto remoto.

D. MPLS

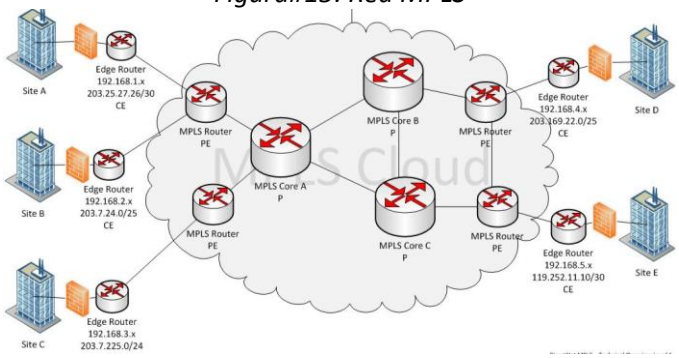
MPLS (Multi Protocol Label Switching) es un mecanismo de alto desempeño ampliamente utilizado en la actualidad por empresas de telecomunicaciones. Definido dentro del RFC 3031, MPLS transmite paquetes de una fuente a un destino remoto (PEs con asignación IP) de un nodo a otro basado únicamente en identificadores cortos denominados etiquetas (labels), en lugar de las tradicionales y largas direcciones IP. Esta característica evita la utilización de una tabla de rutas entre cada nodo, ahorrando tanto rendimiento de los equipos como tiempo de transmisión al no realizar el enrutamiento tradicional. Las etiquetas identifican caminos virtuales (paths) entre los nodos al momento del envío de paquetes. Es importante resaltar que una etiqueta señala el camino completo entre la fuente y el destino, por lo que las etiquetas entre nodos no cambian al momento de transportar un paquete. Además, MPLS soporta la encapsulación de varios protocolos de capa de red, lo que lo hace versátil y aún más atractivo. Este mecanismo ha sido rápidamente aceptado

por las empresas de telecomunicaciones por el impacto que sus características reflejan en los servicios de sus suscriptores, dejando a un lado ahora “viejas” tecnologías como Frame Relay y ATM.

Uno de los principales beneficios de MPLS es el hecho que elimina la dependencia de una tecnología específica de capa 2, (ATM, SONET, Ethernet, Frame Relay) y la necesidad de múltiples redes de capa 2 para satisfacer requerimientos de tráfico. Frame Relay y ATM mueven paquetes a través de frames y celdas a través de la red, respectivamente. Existe una diferencia notoria en el transporte con respecto a MPLS, ya que estas dos tecnologías utilizan un identificador en sus respectivos encabezados, que cambia de un nodo a otro, para identificar cada nuevo salto. Esto implica un mayor rendimiento de la red a diferencia de MPLS.

En la siguiente figura, puede verse un ejemplo común de una red MPLS, en donde las fuentes y los destinos (Routers PE) son identificadas por direcciones IPs, pero los segmentos entre nodos para establecer los caminos entre los PEs, por etiquetas (labels) que identifican el camino completo.

Figura#15: Red MPLS



MPLS está catalogado dentro de la familia de redes tipo Packet-Switching, a pesar de reunir características de ambos tipos. Reúne similitudes a Circuit-Switching, debido a que al momento de la transmisión se crean circuitos “permanentes” entre un nodo y otro, desde la fuente hasta el destino, pero susceptibles a conmutación dependiendo del estado de parámetros definidos y configurables (referentes a degradación y caídas, por ejemplo). Esta última característica y el hecho de que los datos son divididos en paquetes (que no necesariamente llegan en el mismo orden) al momento de transmitir son factores que declinan la balanza por designarlo dentro de la familia Packet-Switching.

Por otro lado, MPLS ha sido generalmente clasificado como un protocolo de “capa 2.5” al reunir características tanto de protocolos de capa 3, por el uso de direcciones IPs entre fuente y destino, como de capa 2 dado el uso de etiquetas (labels) como mecanismo de conmutación (switching). De estos factores y del hecho que es independiente del protocolo de capa 3 utilizado, proviene la designación “Multi Protocol Label Switching”.

A continuación, un resumen de las características de este mecanismo:

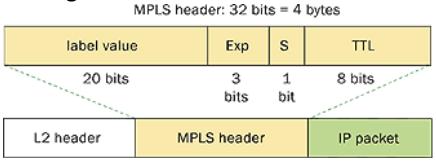
1. CARACTERÍSTICAS

- Packet-Switching: A pesar de cubrir características de Circuit-Switching también.

- Protocolo-independiente: Es independiente del protocolo de capa 3 utilizado y elimina la dependencia de diferentes tecnologías WAN para satisfacer requerimientos de tráfico.
- Alto desempeño: Elimina consumo y reduce el tiempo de transporte en la red al utilizar etiquetas.
- Escalable: Debido al alto desempeño y por ser protocolo-independiente.
- De capa 2.5: Al reunir características de capa 2 y capa 3.
- Conmutación por etiquetas: Transporta de nodo en nodo basado en etiquetas que identifican los distintos caminos entre fuente y destino. Una etiqueta es permanente entre un nodo y otro para señalar un camino específico.

2. ENCABEZADO MPLS. El encabezado de MPLS está compuesto por 32 bits (4 bytes) y dividido en 4 campos. Como puede observarse en la imagen, dicho encabezado forma parte del encabezado general de capa 2, el cual encapsula el paquete IP.

Figura#16: Encabezado MPLS



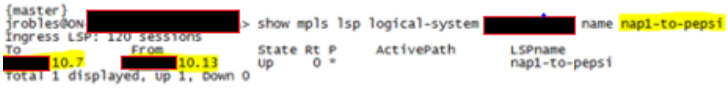
- Valor de etiqueta (Label Value): Campo de 20 bits que representa la 3tiqueta que identifica un camino (path).
- Clase de tráfico (Traffic class): Campo de 3 bits relacionado a Clase de Servicio (Class of Service) o Calidad de Servicio (QoS).
- Bandera de Pila (Stack Flag): Bandera (1 bit) que al estar activo da indicación que la etiqueta es la última en la pila.
- Tiempo de vida (Time To Live): Campo de 8 bits que indica el tiempo de vida de la trama dentro de la red.

3. COMPONENTES

- LSP (Label Switched Path): Representa uno de los conceptos esenciales de MPLS. Es básicamente un túnel unidireccional entre dos Routers (PEs) a través de una red MPLS. En varios aspectos un LSP es similar a un PVC (Permanent Virtual Circuit) usado en Frame Relay o ATM, con la diferencia que un LSP es independiente a la tecnología de capa 2 utilizada. Un LSP es necesario para que la transmisión MPLS ocurra. Es importante resaltar que una etiqueta es utilizada para identificar un LSP entre dos nodos o equipos core (es decir a nivel local), y que mientras un paquete atraviesa una red MPLS, el LSP no cambia.

  En la siguiente figura se muestran los parámetros de un LSP, como lo son la IP fuente y destino, el nombre del LSP y su estado.

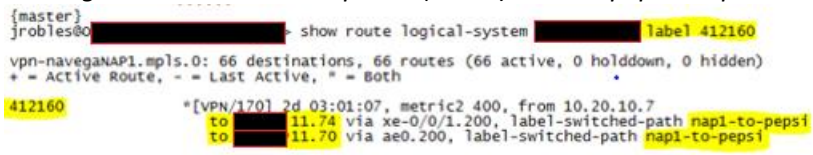
Figura#17: Estado de un LSP en un equipo Juniper



- Label Switch Router (LSR): Router de transito MPLS que realiza el envío de paquetes basado únicamente en el valor de la etiqueta. Este equipo está ubicado en el medio de la red MPLS. Es, en otras palabras, el equipo Core que realiza únicamente tareas de conmutación (switching).

  En la siguiente figura se muestra cómo una etiqueta identifica un LSP en equipos Juniper. Se puede observar el valor de la etiqueta y el nombre del LSP, así como los identificadores de los equipos Core (LSR) que forman parte del LSP, las interfaces de salida, preferencia y tiempo de conocer el path, tal y como se hace en una tabla de rutas.

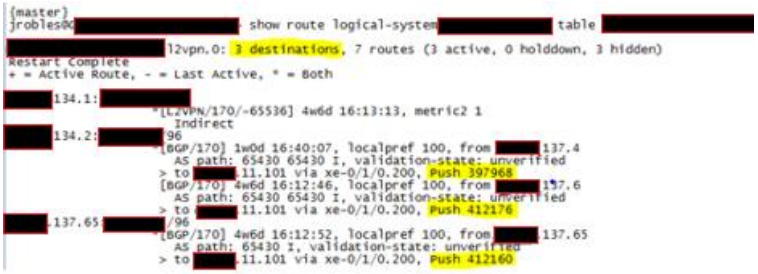
Figura#18: Tabla de etiquetas (labels) en un equipo Juniper



- Label Edge Router (LER): Router borde (PE) de una red MPLS que actúa como la entrada y/o salida de la red. Este equipo encapsula el paquete dentro de un LSP, "empuja" (push) una etiqueta MPLS dentro del mismo si actúa como entrada, y la remueve (pop) si actua como salida. Es importante resaltar que al ingresar paquetes IP dentro de la red, este equipo utiliza información de ruteo para determinar el destino y asignar el valor de etiqueta para hacer referencia al LSP apropiado. Al funcionar como equipo de salida de la red en cambio, remueve la etiqueta y envía el paquete IP fuera de la red MPLS usando reglas normales de ruteo. El destino puede ser bien un equipo CE/CPE o un equipo de una red metro local (en el caso de que la red MPLS sea utilizada a nivel internacional o regional). Dentro de las características de borde que ofrece para la finalización de los servicios a través de la red están L3VPN, L2VPN/Pseudowires y VPLS. (Tanto L3VPN como VPLS se definen más adelante.)

  En la siguiente figura se muestra la cantidad de etiquetas asociadas a un servico VPLS. Se observan los "push" que puede realizar un equipo LER. El "push" identifica la etiqueta. Cabe destacar que un LSP puede ser identificado por una o más etiquetas.

Figura#19: Asignación de etiquetas a un servicio VPLS por parte de un LER en Juniper.

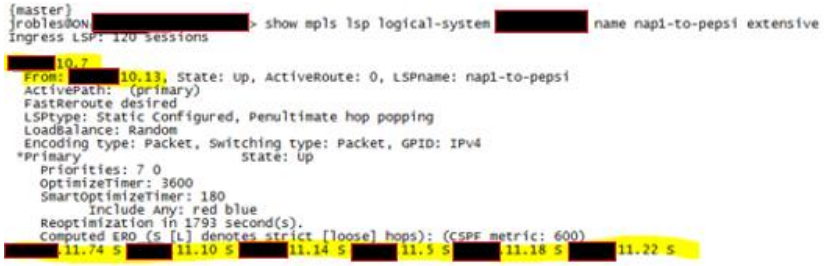


- LDP (Label Distribution Protocol): Protocolo utilizado por LSRs y LERs para la distribución de etiquetas MPLS. Mientras un LSP es un túnel de nivel amplio (al atravesar toda la red) una etiqueta es un identificador local entre dos equipos core. LDP es un protocolo que relaciona

una etiqueta con un LSP entre dichos equipos a través de la red. Es comúnmente utilizado para el transporte de servicios MPLS VPN, entre los que destacan L3VPN (IP-VPN) y VPLS.

En la siguiente figura, se muestra el LSP completo que es identificado por una etiqueta. Cabe destacar que los LSR y LER únicamente necesitan de dicha etiqueta (y no de las direcciones IP que identifican los LSR) para realizar el envío de información.

Figura#20: Identificación extensa de un LSP en equipo Juniper



E. MPLS VPN

Una red privada virtual (VPN, Virtual Private Network) une dos o más equipos de sedes remotas a través de una red de área amplia (WAN), encapsulando los datos y enviándolos de forma privada. MPLS VPN es una familia de métodos que aprovechan los recursos que brindan las redes MPLS para crear VPNs. MPLS VPN puede actuar como una plataforma para el transporte de distintos tipos de servicios, dentro de los que destacan: Multimedia, VoIP, videoconferencia, entre otros. Esto permite reducir la dependencia de las antiguas tecnologías y o técnicas de capa de enlace de datos, como las mencionadas ATM, Frame Relay, Ethernet y SONET. Además MPLS VPN permite a los suscriptores un ahorro económico al contratar servicios por esta vía.

Al ser instalada sobre una red de fibra óptica a nivel de capa física, una red MPLS VPN ofrece una mayor seguridad y ancho de banda, lo que hace este tipo de servicios aún más escalables. Permite una asignación dinámica del ancho de banda dentro de la red MPLS dependiendo del uso de los recursos de la red y de servicios críticos como un tipo de Clase de Servicio. A diferencia de una red Ethernet que, al tener más y más demanda de distintos servicios o usuarios puede presentar caídas o inestabilidad, una red MPLS VPN controla la congestión de forma avanzada por medio de protocolos y algoritmos específicos. Los tipos o familias por medio de los cuales se transmiten datos a través de VPNs se dividen en dos categorías generales, tanto a nivel de capa de red (MPLS L3VPN) como a nivel de capa de enlace de datos (MPLS L3VPN)

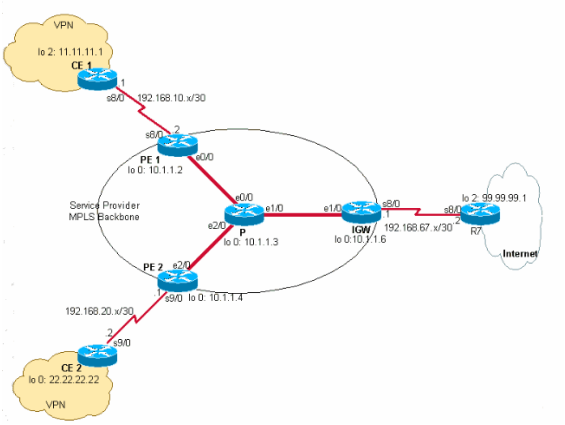
1. MPLS L3VPN. Denominado como BGP/MPLS-VPNs dentro del RFC 2547, se refiere a redes privadas virtuales (VPN) que corren el protocolo IP para comunicación para conexión entre puntos finales dentro de una WAN que corre MPLS. Incluye servicios categorizados dentro de enlaces IP-VPN. Al, dentro de la red WAN, al correr MPLS, entonces se le denomina MPLS IP-VPN o MPLS L3VPN, ofreciendo un desempeño mayor. Este método permite la conexión entre sedes por medio de distintos segmentos de red IP entre los PE, CE y redes internas de los puntos finales, llenando las tablas de rutas de los PE con dichos segmentos de red, por lo que desde la perspectiva del proveedor existe un ruteo a partir de los PE o LER. (Cabe recordar que dentro de una red MPLS no se realizan acciones de ruteo). La privatización de los datos a nivel IP por tipo de servicio o suscriptor normalmente se configura en los equipos PE por medio de las denominadas VRFs (Virtual Routing

and Forwarding) lo que permite la virtualización de los mismos. Es importante mencionar que el protocolo de ruteo utilizado es en la actualidad BGP tanto dentro de la red del proveedor (entre LERs) como entre los equipos PE y equipos de una red Metro o equipos CPE. Mediante este protocolo se comparten las redes entre AS.

Características MPLS L3VPN:

- VPN basada en IP dentro de una red MPLS.
- Creación de VRFs en los LER para la virtualización y privatización de servicios.
- Usualmente las VRFs se clasifican por cliente en donde se reúnen todos sus servicios.
- Soporta topologías distintas y puede interconectar múltiples sitios (es usual de la forma punto-multipunto, donde existe una central hacia varios puntos remotos).
- Conexión a nivel de BGP tanto entre PEs como entre PE-CPE y PE-Metro.
- Normalmente utilizada a nivel empresarial.

En la figura se muestra la topolgía de un enlace de datos IP-VPN dentro de una red MPLS, es decir, que privatiza los datos a través de VRFs y realiza acciones de ruteo fuera de la red, a partir de los LER (PE), segmentando el servicio a nivel IP. En este diagrama, adicionalmente se ofrece un servicio de internet sobre el enlace de datos.



Figura#21: Enlace de datos IP-VPN dentro de una red MPLS

2. MPLS L2VPN. Método asociado a los RFC 4664 y RFC 6136 que permite el transporte de datos entre puntos remotos dentro de una red MPLS y emulando una misma red entre los CPE. La comunicación a través de los puntos remotos puede establecerse por medio de una L2VPN utilizando distintos protocolos de capa de enlace de datos, dentro de los que destaca, por su preferencia dentro de los proveedores de servicio (ISP), Ethernet. Un servicio que emula una conexión Ethernet dentro de una red MPLS se denomina VPLS (Virtual Private LAN Service), el cual se detalla más adelante. MPLS L2VPN permite la entrega del servicio a los suscriptores de forma transparente, lo que significa que a nivel de capa 3 únicamente existe un segmento de red IP y es manejado por dicho suscriptor. Para el proveedor del servicio (ISP), se aprecia como una conexión de capa 2, a nivel de conmutación (switcheo), en el que se identifican las MAC de los equipos remotos que son levantados a capa 3 (usualmente los equipos del suscriptor). Al igual que una MPLS L3VPN, los datos también se privatizan por suscriptor manejando las interconexiones de las sedes remotas implicadas dentro de una instancia de ruteo (Routing-instance) pero del tipo o familia VPLS en lugar de VRF. El hecho de que para el suscriptor sea un servicio transparente, haciendo uso de

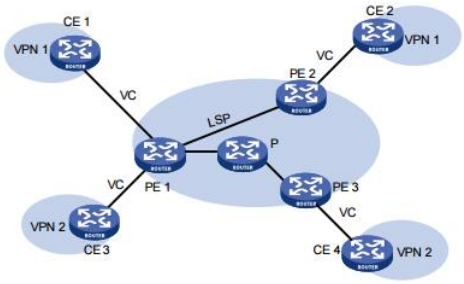
un único segmento de red para la comunicación remota, ha hecho que hoy en día sea una opción preferente dentro del mercado, en particular el servicio VPLS. A continuación, se presenta un breve resumen de las características de un servicio L2VPN del tipo VPLS dentro de una red MPLS.

Características MPLS L2VPN (VPLS):

- Emula una conexión Ethernet multipunto a través de MPLS.
- Utilizado para enlazar varios puntos remotos dentro del mismo segmento de red o dominio de broadcast.
- Evita la necesidad de la configuración de L2circuits dentro de los equipos para emular la conexión de capa 2.
- Emula las características de un switch de capa 2 (Aprendizaje de MACs, decisiones de envío/filtrado).

En la siguiente figura, se muestra un diagrama de un servicio L2VPN VPLS a través de una red MPLS. La red MPLS puede ser conecibda como un switch en donde los CPE son los hosts que se conectan a través de los puertos (PE) del mismo.

Figura#22: Enlace de datos VPLS dentro de una red MPLS.



F. BGP

Definido dentro del RFC 4271, es un protocolo del tipo EGP (Exterior Gateway Protocol) que permite la comunicación a través de equipos borde de dos sistemas autónomos (AS). Esto permite compartir las tablas de rutas entre dichos sistemas al momento de establecerse una sesión BGP. Ha sido clasificado como un protocolo vector-camino (path-vector), pero también como un protocolo vector-distancia (distance-vector). BGP determina decisiones de ruteo a través del estado de los caminos (paths), políticas de red, o reglas configuradas por el administrador de la red. Puede ser utilizado para la interconexión de dos equipos remotos dentro de un mismo AS. En ese sentido es referido como iBGP (interno). Este protocolo es definido como el que permite el Internet al interconectar distintos ISP a través de sus sistemas autónomos. Es bastante complejo y difícil de configurar, sin embargo la seguridad y escalabilidad que ofrece lo hace esencial hoy en día.

Los equipos que utilizan BGP hacen uso del protocolo TCP para establecer la comunicación y envían información actualizada de sus tablas de rutas únicamente al detectar un cambio. Las tablas de rutas contienen una lista de redes que pueden alcanzar a través de sus interfaces de salida, tiempo de conocerlas, preferencia y una métrica de costo asociada al camino hacia cada router de tal manera que la mejor ruta sea siempre escogida. Al detectar un cambio en la elección de rutas óptimas, únicamente la parte afectada de la tabla es enviada.

1. TERMINOLOGÍA BGP

- AS: Sistema autónomo, infraestructura de red manejada por una misma entidad y que posee su propio conjunto de políticas de red. Usualmente los suscriptores finales pertenecen al AS de un ISP, sin embargo los suscriptores que re-venden los servicios poseen su propio AS y la conexión entre ambos es típicamente hoy en día a través de BGP.

- Número de AS: Identificador único para cada AS utilizando en el proceso de reconocimiento entre vecinos BGP (peers).

- Camino de AS: Conocido como AS-Path, es el camino completo de un paquete que lista los sistemas autónomos que debe atravesar para llegar a su destino.

- BGP: Protocolo de salida de borde (Border Gateway Protocol).

- EGP: Protocolo de salida de exterior (Exterior Gateway Protocol). Es importante mencionar que BGP es un EGP, pero también existe un antiguo protocolo llamado EGP.

- Peer: Equipo vecino (neighbor) y borde en donde se configura BGP para su interconexión con el vecino de otro AS.

- IGP: Protocolo de salida de interior (Interior Gateway Protocol). Utilizado para la interconexión de una infraestructura de red de un ISP.

- Multihoming: Término utilizado cuando existe una interconexión entre AS. BGP permite compartir por defecto las tablas de rutas entre equipos borde siempre y cuando conozca las redes por BGP. El administrador de la herramienta debe configurar políticas de ruteo para compartir rutas conocidas por medio de otros protocolos.

- Política de red: Conjunto de enunciados en los que se determina la forma de importar y exportar rutas a través de un protocolo de ruteo. En este caso, BGP.

2. OPERACIÓN. La conexión BGP entre peers es establecida por configuración manual entre equipos de capa 3 para crear una sesión TCP por el puerto 179. De tal forma que el protocolo tome las decisiones apropiadas entre sus vecinos, se reconocen seis estados distintos dentro de la operación del mismo:

- Estado Libre (Idle): se inicia la comunicación TCP entre equipos vecinos y establece una conexión segura. En este estado se rechazan todas las conexiones BGP como tal. Cambia a estado Conexión (Connect) y si un error ocurre regresa al estado Libre.

- Estado Conexión (Connect): Espera a una negociación exitosa vía TCP. Si la sesión es exitosa envía el Mensaje de Apertura (Open Message) para cambiar al estado Envío-Apertura (OpenSent). Las razones por las que puede existir un error y volver al estado Libre pueden ser entre otras que el puerto TCP 179 no esté habilitado, dirección IP incorrecta entre Peers, incorrecta asignación del número de AS.

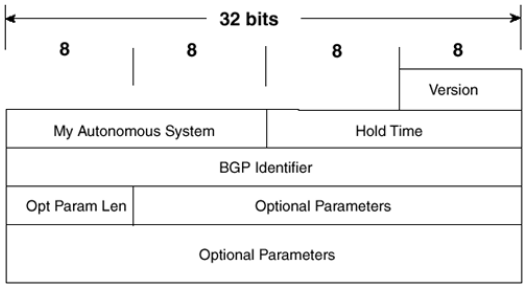
- Estado Activo (Active): Ocurre si un equipo vecino no pudo establecer una sesión TCP exitosa. En este estado se intenta nuevamente establecer la sesión para enviar el Mensaje de Apertura. Si vuelve a falla, regresa el estado Libre. Fallas repetidas pueden causar un ciclo entre estados Libre y Activo. Las causas entre otros pueden deberse a configuración errónea de la sesión BGP, congestión en la red e intermitencias en las interfaces de conexión.

- Estado Envío-Apertura (OpenSent): El vecino escucha el Mensaje de Apertura (OpenSent) de su contraparte y al recibirlo lo valida. Si no es exitoso por alguna discrepancia en los parámetros de configuración, se envía un mensaje de error. De lo contrario, se pasa al estado Confirmación-Apertura.

- Estado Confirmación-Apertura (OpenConfirm): Si el tiempo de respuesta no se vence antes de recibir el mensaje de confirmación, la sesión se establece, cambiando al estado final (Established). De lo contrario, regresa al estado Libre.

- Estado Establecimiento (Established): Los equipos vecinos se intercambian mensajes de actualización para compartir las redes dentro de sus tablas de rutas. Si ocurre algún error en los mensajes de actualización o si se vence el tiempo de respuesta, se regresa al estado Libre (Idle).

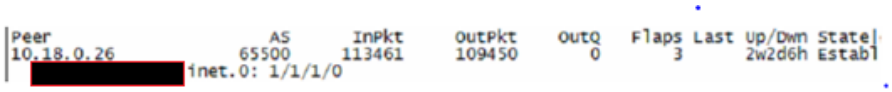
A continuación, se muestra el formato y una explicación corte de los campos de interés del mensaje OpenSent.



Figura#23: Formato del mensaje OpenSent

- Versión: Campo de un byte que identifica la versión del BGP.
- Sistema autónomo: Campo de dos bytes para identificar el número de AS.
- Identificador BGP: Router ID (4 bytes) del equipo fuente o vecino que envía el mensaje.
- Tiempo de espera (hold time): De dos bytes, número de segundos de espera previo a rechazar la conexión.

En la siguiente figura, se muestra el estado de una sesión BGP en un equipo Juniper.



Figura#24: Sesión BGP en un equipo Juniper

G. VPLS (VIRTUAL PRIVATE LAN SERVICE)

Es un servicio LAN privado virtual basado en Ethernet y de conexión punto-multipunto. Permite la conectividad remota de sedes a través de una red MPLS emulando una red Ethernet. Está definido dentro de los RFC 4761 (señalización BGP a través de la MPLS) y RFC 4762 (señalización LDP a través de la MPLS). El hecho que emule una conexión Ethernet quiere decir que a nivel del suscriptor todos sus puntos aparentan establecer conectividad a través de un mismo dominio de difusión (broadcast) aunque en realidad atraviesan toda una red WAN para este fin. En su configuración e implementación, VPLS es bastante similar a una implementación L2VPN. En VPLS, un paquete que es originado desde una de las sedes del suscriptor debe primero pasar por un equipo CE, que bien puede ser un router o un switch. Luego, es enviado a un equipo borde de entrada (PE, LER) el cual le da entrada a la red MPLS. Para atravesar dicha red, el paquete es asociado a un LSP por medio de una etiqueta y así distribuido entre los equipos LSR, hasta llegar a un equipo borde de salida (PE, LER). Por último es enviado al CE en el punto remoto, usualmente a través de una red local (Metro o MPLS), para llegar a su destino (red remota de suscriptor).

Una de las principales diferencias entre VPLS y otros servicios del tipo MPLS L2VPN, es que por medio de VPLS, una sede puede difundir (hacer un broadcast) a todas sus sedes remotas, tal y como un switch haría a través de todos sus puertos (excepto del cual lo recibió) al identificar un mensaje de broadcast. Las conexiones MPLS L2VPN típicas son estrictamente punto a punto. VPLS es punto-multipunto. Como se ha mencionado anteriormente, los proveedores del servicio suelen privatizar los enlaces en función de cada suscriptor y las sedes que lo conforman dentro de una instancia de ruteo (routing-instance) del tipo vpls. Todos los caminos de interconexión y que transportan tráfico entre los equipos borde (PEs) que conforman el servicio VPLS y que participan dentro de una instancia de ruteo son denominados Pseudowires. La señalización entre los mismos puede ocurrir por BGP o LDP.

1. RUTEO VPLS. La afirmación de que un servicio VPLS tiene el comportamiento de una red Ethernet obliga a encontrar una similitud con una red típica de ese tipo. Como se ha mencionado con anterioridad, un servicio VPLS puede ser mapeado a gran escala como un switch virtual al cual se conectan sus hosts y por medio del cual intercambian información a través de sus puertos, en donde el mapeo respectivo se puede ver de la siguiente manera.

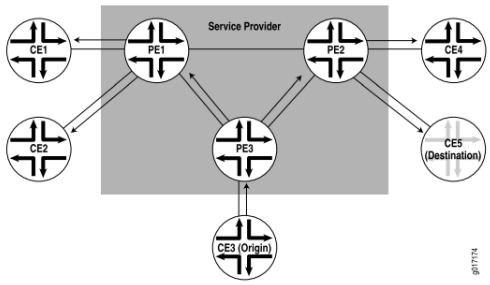
- Switch: Representa la red MPLS sobre la cual se establece el servicio.
- Puertos de conexión de switch: Representa los equipos bordes por medio de los cuales se permite el ingreso a la red MPLS. En este caso la comparativa difiere en cierto sentido, ya que varios CE pueden conectarse por el mismo PE, es decir varios “hosts” a un mismo “puerto”.
- Hosts: Se refiere usualmente a los CE o equipos terminales en las sedes remotas que intercambian información a nivel de capa 3. En realidad, es cualquier equipo dentro del servicio VPLS que levanta una conexión de capa 3. Esto puede ser visto del lado del proveedor con fines de monitoreo aplicado desde equipos en su red MPLS y reflejado en la tabla MAC de la VPLS.

Cuando un equipo PE configurado por una instancia de ruteo VPLS recibe un paquete de un dispositivo CE, primero determina si conoce el destino. Si es así, lo envía a través de la red MPLS hasta el PE correspondiente, el cual se encarga de hacerlo llegar hasta el CE remoto. Si no conoce el destino, el PE envía el paquete a todos los CE a través de los PE, esperando que alguno responda

con su MAC (en el caso de que el CE sea un router que levanta la conexión IP) o con la MAC del equipo correspondiente conectado al CE dentro la sede remota destino (en el caso de que el CE sea un switch) para que el PE pueda asociar dicha MAC dentro de la tabla MAC de la VPLS, y así conocer la ruta hacia el destino. En cualquiera de ambos casos, el dispositivo que envía el paquete siempre es distinto al que lo recibe. Por otra parte, cuando un PE recibe un paquete de otro dispositivo PE, revisa el destino. Si el destino (CPE) corresponde al sitio local, entonces lo envía a dicho dispositivo. Si el destino corresponde a un sitio remoto entonces descarta el paquete.

En la siguiente figura, se muestra el proceso de difusión (broadcast o flooding) en un servicio VPLS cuando el PE que recibe el paquete del CPE no conoce el destino al no tenerlo asociado en la tabla MAC.

Figura#25: Proceso de "Flooding" en un servicio VPLS

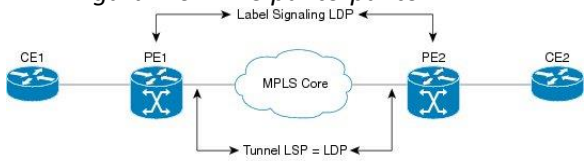


Con la finalidad de implementar un sistema de monitoreo que indique una alerta al observar un comportamiento anómalo que pueda sugerir la caída del servicio en uno o más puntos remotos, es necesario comprender cuatro partes fundamentales de un servicio VPLS: su topología, su configuración, sus conexiones y su tabla MAC. Esto debido a que el monitoreo debe cubrir todos los posibles escenarios que puede presentar una red VPLS para que sea escalable, eficaz y adaptable. Para fines de este trabajo, estas partes son explicadas desde el punto de vista de equipos Juniper.

2. TOPOLOGÍAS DE UN SERVICIO VPLS (EN GENERAL). A continuación se muestran los posibles escenarios de una red VPLS sobre MPLS:

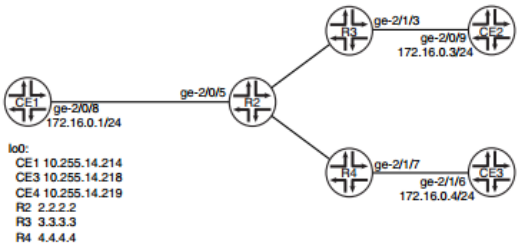
- Punto a punto: Este escenario ocurre cuando se tiene únicamente dos CE, interconectados por dos o más PE (si hay más de dos, existe redundancia por equipo), dentro de la red MPLS.

Figura#26: VPLS punto-punto

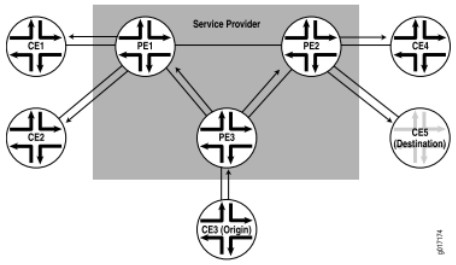


- Punto-Multipunto: Escenario cuando la conexión se da entre más de dos CE. Se tienen varios escenarios. Por ejemplo cuando se tiene más de dos PE, pero con un único CE asociado por PE. O bien, cuando se tiene más de dos PE, pero con más de un CE asociado por PE. En estos escenarios, también un CE puede tener uno o más PE (redundancia de equipo).

Figura#27: VPLS punto-multipunto (1)

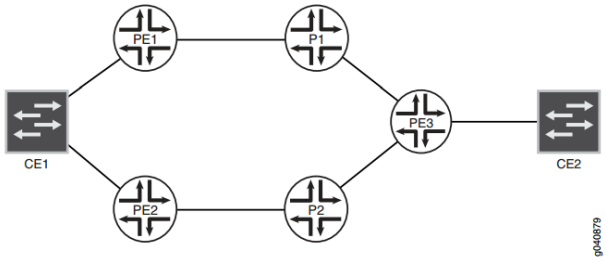


Figura#28: VPLS punto-multipunto (2)



- Redundante (Multihomed): Recurso que puede darse en cualquiera de los tres escenarios presentados con anterioridad. La redundancia puede darse de dos formas, a nivel de equipo o a nivel de interfaz. A nivel de equipo es cuando se provee redundancia para un CE, el cual posee más de un PE asociado. A nivel de interfaz, es cuando para un CE, existe un solo PE pero con más de una interfaz asociada hacia el CE. Para la figura mostrada, la redundancia se presenta en el escenario punto a punto y a nivel de equipo.

Figura#29: VPLS con redundancia a nivel de equipo



3. CONFIGURACIÓN DE UN SERVICIO VPLS (EN JUNIPER). Dentro de las configuraciones de VPLS, los CE que se interconectan pueden ser routers o switches. Si son switches y hay múltiples conexiones entre un PE y un CE (redundancia de interfaz) es importante habilitar el protocolo Spanning Tree para evitar bucles no deseados (loop avoidance). Para que una VPLS sea funcional, la configuración que debe ocurrir en los PE puede resumirse de la siguiente manera:

- Habilitar la funcionalidad VPLS.
- Distribución de información de ruteo entre PE.
- Configuración de circuitos virtuales PE-PE y PE-CE.

Cada conexión VPLS debe configurarse dentro de una instancia de la familia vpls. De esta forma se garantiza una red transparente por el suscriptor a través de la red MPLS. Por otro lado, todas las

interfaces virtuales de conexión de un PE hacia uno o más CE locales deben ser configuradas dentro de la instancia del tipo vpls. Debe también realizarse la configuración de los LSP entre equipos PE. En la siguiente figura, se muestra toda la configuración posible de una instancia del tipo vpls. Es necesario recalcar que pueden omitirse sentencias lo que usualmente provoca que el equipo escoja un parámetro por defecto asociado a las mismas.

Figura#30: Configuración de una instancia vpls

```
vpls {
  active-interface {
    any;
    primary interface-name;
  }
  connectivity-type (ce | irb | permanent);
  encapsulation-type encapsulation-type;
  interface-mac-limit limit;
  label-block-size size;
  mac-table-aging-time time;
  mac-table-size size;
  neighbor neighbor-id;
  no-tunnel-services;
  site site-name {
    active-interface {
      any;
      primary interface-name;
    }
    interface interface-name {
      interface-mac-limit limit;
    }
    mesh-group mesh-group-name;
    multi-homing;
    site-identifier identifier;
    site-preference preference-value;
  }
  site-range number;
  traceoptions {
    file filename <files number> <size size> <world-readable | no-world-readable>;
    flag flag <flag-modifier> <disable>;
  }
  tunnel-services {
    devices device-names;
    primary primary-device-name;
  }
  vpls-id vpls-id;
}
```

Existen dos formas de señalizar el tráfico dentro de un servicio VPLS, a través de BGP y por medio de LDP. El equipo comprende una u otra señalización dependiendo del tipo de sentencias configuradas. Es importante mencionar que no es posible establecer los dos tipos de señalización para una misma instancia vpls. Si se trata de configurar sentencias de ambos tipos, el equipo da un error al momento de establecer la configuración.

A continuación, se muestra las sentencias que determinan una u otra señalización.

Señalización BGP

- *site*
- *site-identifier*
- *site-range*

Señalización LDP

- *neighbor*
- *vpls-id*

A continuación, se detalla una breve explicación de las sentencias dentro de la configuración.

- *active-interface:* establece una redundancia por interfaz.
  - *any:* se escoge aletoriamente la interfaz activa.
  - *primary interface-name:* se escoge explícitamente la interfaz activa.

- *connectivity-type:* determina cuando la conexión de una VPLS debe caerse dependiendo del parámetro de interfaz asignado. Por defecto, ce.
  - *ce:* parámetro que establece que cuando la interfaz VPLS directamente conectada a alguna interfaz de red local pierde conectividad, la conexión de la VPLS se cae.
  - *irb:* si una interfaz del tipo irb está configurada dentro del servicio, la VPLS perderá la conectividad únicamente si dicha interfaz se cae.
  - *permanent:* la conexión se pierde sólo si todas las interfaces lógicas asociadas se caen.

- *encapsulation-type:* el tipo de encapsulación para los pseudowires establecidos entre PE. Por defecto, *ethernet.*
  - *ethernet:* Ethernet.
  - *ethernet-vlan:* Ethernet Virtual LAN.

- *interface-mac-limit:* establece un límite para direcciones MAC aprendidas de interfaces locales.

- *label-block-size:* determina el tamaño del bloque interno de etiquetas. En MPLS VPLS, se maneja una pila de dos etiquetas. La externa, normalmente utilizada dentro de la red MPLS, y la interna, propia de VPLS. Una etiqueta interna es necesaria por cada sitio (PE) en la red VPLS. Por defecto, 8.
  - *2:* maneja las etiquetas en incrementos de dos. Usual para una VPLS con dos sitios (PE).
  - *4:* maneja las etiquetas en incrementos de 4.
  - *8:* maneja las etiquetas en incrementos de 8.
  - *16:* maneja las etiquetas en incrementos de 16.

- *mac-table-aging-time:* establece el tiempo en segundos en el que una dirección MAC es removida de la tabla si no recibe cualquier tipo de tráfico proveniente de dicha dirección. Valores desde 10 hasta un millón de segundos. Por defecto, 300. A partir de la versión 8.4, esta opción no está habilitada en equipos de la serie MX en la jerarquía *[edit routing instances protocols vpls]*, sino que únicamente se puede aplicar al equipo completo en la jerarquía *[edit protocols l2-learning]*.
  (http://www.juniper.net/documentation/en_US/junos12.3/topics/task/configuration/vpls-mac-address-timeout-solutions.html)

- *mac-table-size:* permite establecer la cantidad de direcciones MAC que pueden ser aprendidas. Mínimo 16, máximo 65536 y por defecto, 512.

- *neighbor:* especifica el identificador único de un PE dentro de una VPLS señalizada por LDP.

- *no-tunnel-services:* permite identificar las conexiones por medio de interfaces lógicas remotas del tipo LSI (Label-switched interfaces) en lugar de una interfaz de túnel virtual.

- *site site-name:* asigna un nombre a un sitio dentro de una VPLS. El término sitio se refiere explícitamente a un PE dentro de la instancia.
    - *active-interface:* esta opción se incluye dentro de esta jerarquía al especificar señalización por BGP.
    - *interface*
        interface-mac-limit: utilizada dentro de la jerarquía al especificar señalización por BGP.
    - *mesh-group mesh-group-name:* asocia la interconexión entre pseudowires dentro de un grupo específico.
    - *multihoming:* establece redundancia a nivel de equipo.
    - *site-identifier:* asigna un identificador único y numérico al sitio VPLS.
    - *site-preference:* valor numérico que determina la preferencia del sitio VPLS. Usualmente utilizado si *multihoming* es especificado.
        - *primary:* referido al valor numérico 65,536, siendo así el sitio más preferente.
        - *backup:* valor de 1, indicando que el sitio es el menos preferente.

- *site-range number:* número entre 1 y 65,534 para especificar el límite superior de la cantidad de sitios de una VPLS. Este valor debe ser mayor al de cualquier *site-preference* dentro de la red. Por defecto, 65,534.

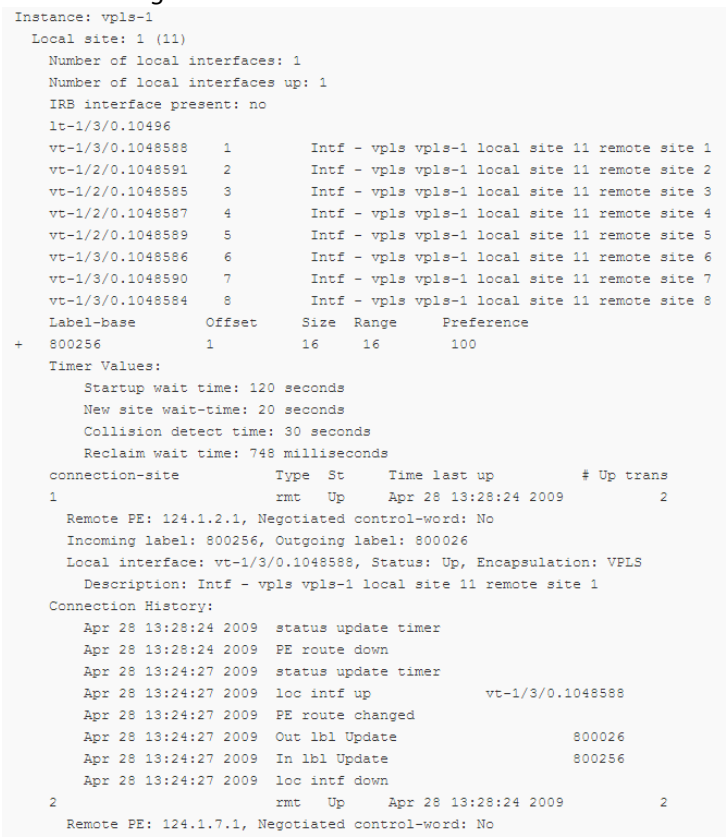
- *traceoptions:* opción habilitada para debugging.

- *tunnel-services:* especifica que el tráfico sea transmitido por una interfaz del tipo túnel virtual (VT). Esto permite el balanceo del tráfico a través de todas las interfaces VT disponibles para el PE.
    - *devices device-names:* especifica las interfaces VT a utilizarse para la instancia. Si esta opción no se específica, todas las interfaces VT disponibles para el equipo se habilitan.
    - *primary primary-device-name:* especifica la interfaz VT primaria para el uso dentro de la instancia.

- *vlan-id:* especifica el identificador único para la instancia VPLS.

4. CONEXIONES EN UN SERVICIO VPLS (EN JUNIPER). Una vez se ha realizado una configuración exitosa entre los PE que conforman el servicio VPLS, las conexiones entre los distintos sitios se establecen. La figura a continuación (no mostrada en su totalidad), muestra el estado de las conexiones de un servicio VPLS vista desde un PE, cuyos aspectos más importantes se detallan.

Figura#31: Conexiones de una red VPLS



```
Instance: vpls-1
  Local site: 1 (11)
    Number of local interfaces: 1
    Number of local interfaces up: 1
    IRB interface present: no
    lt-1/3/0.10496
    vt-1/3/0.1048588    1         Intf - vpls vpls-1 local site 11 remote site 1
    vt-1/2/0.1048591    2         Intf - vpls vpls-1 local site 11 remote site 2
    vt-1/2/0.1048585    3         Intf - vpls vpls-1 local site 11 remote site 3
    vt-1/2/0.1048587    4         Intf - vpls vpls-1 local site 11 remote site 4
    vt-1/2/0.1048589    5         Intf - vpls vpls-1 local site 11 remote site 5
    vt-1/3/0.1048586    6         Intf - vpls vpls-1 local site 11 remote site 6
    vt-1/3/0.1048590    7         Intf - vpls vpls-1 local site 11 remote site 7
    vt-1/3/0.1048584    8         Intf - vpls vpls-1 local site 11 remote site 8
    Label-base        Offset     Size  Range     Preference
+   800256            1          16    16        100
    Timer Values:
        Startup wait time: 120 seconds
        New site wait-time: 20 seconds
        Collision detect time: 30 seconds
        Reclaim wait time: 748 milliseconds
    connection-site           Type  St     Time last up          # Up trans
    1                         rmt   Up     Apr 28 13:28:24 2009           2
      Remote PE: 124.1.2.1, Negotiated control-word: No
      Incoming label: 800256, Outgoing label: 800026
      Local interface: vt-1/3/0.1048588, Status: Up, Encapsulation: VPLS
        Description: Intf - vpls vpls-1 local site 11 remote site 1
    Connection History:
        Apr 28 13:28:24 2009  status update timer
        Apr 28 13:28:24 2009  PE route down
        Apr 28 13:24:27 2009  status update timer
        Apr 28 13:24:27 2009  loc intf up             vt-1/3/0.1048588
        Apr 28 13:24:27 2009  PE route changed
        Apr 28 13:24:27 2009  Out lbl Update                    800026
        Apr 28 13:24:27 2009  In lbl Update                     800256
        Apr 28 13:24:27 2009  loc intf down
    2                         rmt   Up     Apr 28 13:28:24 2009           2
      Remote PE: 124.1.7.1, Negotiated control-word: No
```

- Instancia (instance): Indica el nombre de la instancia vpls.

- Sitio local (local site): Nombre del sitio local que identifica al PE. Entre paréntesis, se indica el identificador numérico del mismo (site-identifier).

- Número de interfaces locales: Identifica la cantidad de interfaces locales en el sitio. No debe considerarse necesariamente proporcional al número de CE asociado al PE debido a que puede existir interfaces por redundancia.

- Número de interfaces locales activas: Identifica la cantidad de interfaces locales activas en el sitio local.

- Presencia de interfaces IRB: Identifica la presencia interfaces del tipo IRB configuradas dentro de la instancia.

- Interfaces: Listado de las interfaces lógicas locales activas primarias y de redundancia, y las interfaces lógicas remotas. Se debe resaltar que sólo se hace distinción entre las interfaces lógicas remotas por sitio remoto. Es decir que si un sitio remoto tiene tres sedes, a nivel del sitio local las tres se identifican por la misma LSI. Ahora si existen dos sitios remotos, entonces sí se diferencian por dos LSI, desde el sitio local. En este apartado, pueden verse interfaces del tipo VT o LSI a nivel remoto. (Esto se comprobó en el diseño experimental.) Para las interfaces locales dentro del listado únicamente se especifica su nombre. Para las

interfaces remotas dentro del mismo, se especifica el nombre en adición a los siguientes detalles:
  - Identificador número del sitio (site-identifier) remoto del cual proviene.
  - Nombre de la instancia vpls a la que pertenece.
  - Descripción que incluye el identificador del número de sitio (site-identifier) local del PE seguido del identificador remoto.

- Etiqueta base (Label-base): Primera etiqueta del bloque de etiquetas. Es la etiqueta utilizada por un PE al momento de enviar información a través de la red.

- Offset: Valor usado para identificar el pseudowire establecido para la conexión entre dos PE.

- Tamaño (size): Se refiere al tamaño del bloque de etiquetas (label block size).

- Rango (range): Rango del bloque de etiquetas.

- Preferencia (preference): Preferencia del sitio local. De utilidad cuando existe redundancia a nivel de equipo.

Por cada punto de conexión remota establecido, destacan los siguientes aspectos:

- Sitio de conexión (connection-site): Es el identificador numérico del sitio remoto.

- Tipo (type): Tipo de conexión, local o remota.

- Estado (St): Estado de la conexión. Destaca Up (arriba) y Dn (abajo).

- Último tiempo de conexión (Time last Up): Fecha del último establecimiento de la conexión.

- Número de transiciones (# Up trans): de estado Dn a estado Up.

- PE Remoto (Remote PE): Dirección IP utilizada para identificar al PE remoto.

- Etiqueta de entrada (Incoming label): Nombre de la etiqueta de entrada. Suele ser la etiqueta base del sitio local.

- Etiqueta de salida (Outgoing label): Nombre de la etiqueta de salida. Suele ser la etiqueta base del sitio remoto.

- Interfaz local (Local interface): Se refiere al nombre de la interfaz local (LSI o VT) para la conexión remota.

- Estado (Status): Estado de la interfaz local del punto remoto.

- Encapsulación: Encapsulación de la interfaz proveniente del punto remoto.

- Descripción: Incluye el identificador del número de sitio (site-identifier) local del PE seguido nuevamente del identificador remoto.

En la figura a continuación, se muestra el conjunto de leyendas para el estado de una VPLS y de una interfaz virtual dentro de la VPLS.

Figura#32: Listado de leyendas sobre el estado de conexión de una VPLS y sus interfaces

```
Legend for connection status (St)
EI -- encapsulation invalid      NC -- interface encapsulation not CCC/TCC/VPLS
EM -- encapsulation mismatch     WE -- interface and instance encaps not same
VC-Dn -- Virtual circuit down    NP -- interface hardware not present
CM -- control-word mismatch      -< -- only outbound connection is up
CN -- circuit not provisioned    >- -- only inbound connection is up
OR -- out of range               Up -- operational
OL -- no outgoing label          Dn -- down
LD -- local site signaled down   CF -- call admission control failure
RD -- remote site signaled down  SC -- local and remote site ID collision
LN -- local site not designated  LM -- local site ID not minimum designated
RN -- remote site not designated RM -- remote site ID not minimum designated
XX -- unn connection status  IL -- no incoming label
MM -- MTU mismatch               MI -- Mesh-Group ID not availble
BK -- Backup connection          ST -- Standby connection
PF -- Profile parse failure      PB -- Profile busy

Legend for interface status
Up -- operational
Dn -- down
```

5. TABLA DE DIRECCIONES MAC DE UN SERVICIO VPLS (EN JUNIPER). Una tabla de direcciones MAC de una conexión VPLS muestra el conjunto de direcciones MAC aprendidas tanto a nivel local como remoto. Las interfaces locales suelen ser subinterfaces de interfaces físicas comunes (giga, 10giga, fast, etc). A nivel remoto, las interfaces pueden ser del tipo VT o LSI como ya se ha explicado con anterioridad. Es importante mencionar que al aprender MACs de los puntos, la VPLS captura una etiqueta de salida para identificar y alcanzar al sitio remoto y una etiqueta de entrada, para tráfico recibido de dicho sitio remoto. En el caso de no configuración de túneles virtuales, la etiqueta de salida es utilizada para identificar a la interfaz del tipo LSI proveniente del PE remoto, de los CE asociados. Una restricción importante al momento de la difusión de paquetes (flooding) dentro de la red VPLS es que el tráfico recibido de PE remotos, nunca es enviado a otros PE. Esto ayuda a prevenir bucles no deseados. Sin embargo, si un switch CE tiene una o más conexiones a un PE específico (redundancia por interfaz), es necesario habilitar el protocolo Spanning Tree en dicho CE para evitar los mencionados bucles.

Se dice que un servicio otorgado por VPLS es transparente en el sentido que todos los dispositivos del suscriptor están ubicados lógicamente dentro de la misma red de difusión (broadcast). Esto quiere decir que es únicamente el suscriptor el que levanta conexiones a nivel de capa de red desde cada una de sus sedes. Sin embargo, para fines de un monitoreo fácil y más eficiente puede ocurrir que el proveedor del servicio levante este tipo de conexiones IP (usualmente con el apoyo del suscriptor) entre algún equipo dentro de la red del mismo y los equipos del suscriptor. Esto crea otro dominio de difusión (broadcast) haciendo un tanto de menos el concepto de transparencia.

Es muy importante tener claro que todas las direcciones MAC de las interfaces físicas en donde se levanten conexiones de capa de red, aparecerán en la tabla MAC de la VPLS. Un CE puede ser un switch o un router. Si es un switch, todas las direcciones MAC de las interfaces físicas de los equipos del cliente que se conecten al mismo (y levanten conectividad IP), aparecerán en la tabla MAC. Esto puede ser un tanto ineficiente desde el punto de vista del proveedor, debido a que la tabla MAC de

la VPLS de los PE se llenará de direcciones por cada sede (donde el CE sea un switch), y al mismo le bastaría con conocer una sola dirección MAC por sede para saber que su transmisión se encuentra operativa. Es por ello que muchos proveedores optan por instalar un CE que sea un router, en donde el mismo levante una conexión IP y así evitar y limitar la transmisión de las direcciones MAC de todos los equipos conectados al mismo. En este caso, únicamente se transmitiría la MAC de la interfaz física del CE conectada al PE. En el caso de que el suscriptor revenda el enlace, se espera la colaboración del mismo para instalar un router como CE, ya que es usual que el CE no pertenezca al proveedor.

En la siguiente figura se muestra las tablas MAC de un PE por instancia vpls. En este punto se debe mencionar que para que una VPLS se considere operacional en su totalidad, debe reconocerse al menos una dirección MAC por cada sede remota (por CE), a través de las interfaces lógicas remotas. Como ya se mencionó, a nivel local sí se puede hacer distinción de cantidad de interfaces por sitio, pero a nivel remoto, todas las interfaces de un mismo sitio se identifican por la misma LSI, desde el punto de vista del PE local. En este ejemplo ninguna de las dos instancias vpls se consideraría operacional desde el punto de vista del proveedor.

Figura#33: Tabla de direcciones MAC por instancia vpls

```
MAC flags (S -static MAC, D -dynamic MAC,
           SE -Statistics enabled, NM -Non configured MAC)

Routing instance : vpls_ldp1
VLAN : 223
   MAC                 MAC      Logical
   address             flags    interface
   00:90:69:9c:1c:5d   D        ge-0/2/5.400

MAC flags (S -static MAC, D -dynamic MAC,
           SE -Statistics enabled, NM -Non configured MAC)

Routing instance : vpls_red
VLAN : 401
   MAC                 MAC      Logical
   address             flags    interface
   00:00:aa:12:12:12   D        lsi.1051138
   00:05:85:74:9f:f0   D        lsi.1051138
```

- Instancia de ruteo (routing-instance): Se refiere al nombre de la instancia vpls a la que la tabla de direcciones MAC corresponde.

- VLAN: Es el identificador de la LAN virtual sobre la cual se configuran las subinterfaces de la VPLS.

- Dirección MAC (MAC address): Muestra el conjunto de direcciones MAC reconocidas a nivel local y remoto por medio de las interfaces lógicas.

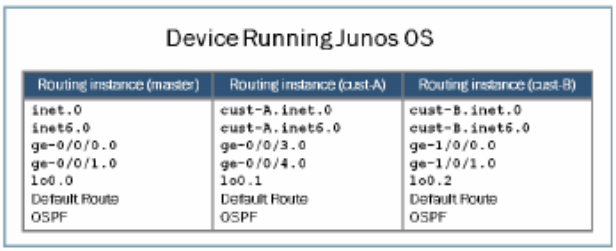
- Banderas MAC (MAC flags): Indica el estado de las direcciones MAC aprendidas por medio de cada interfaz lógica.
    - S: Mac Address configurada estáticamente en la tabla.
    - D: Dirección MAC dinámica.
    - SE: Registro de direcciones MAC habilitado.
    - NM: Mac no configurada.

- Interfaz lógica (Logical interface): Indica el nombre de la interfaz lógica de la cual se aprende cada dirección MAC.

H. INSTANCIA DE RUTEO EN JUNIPER (ROUTING INSTANCE)

Una instancia de ruteo es una virtualización de un equipo físico de tal forma que todas sus propiedades sean independientes de otras instancias y de la configuración fuera de las mismas. Esto permite emular varios equipos dentro de un mismo equipo físico permitiendo una gran flexibilidad del dispositivo. Una instancia de ruteo es una colección única y lógica de tablas de ruteo, interfaces y parámetros de protocolos de ruteo. En la figura a continuación se ejemplifica la virtualización dentro de un equipo físico a través de estas instancias. Incluso una misma interfaz física, a través de las interfaces lógicas derivadas de ella, puede formar parte de distintas instancias de ruteo.

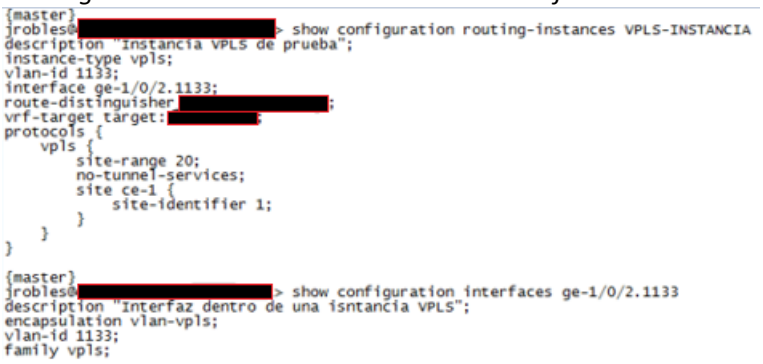
Figura#34: Virtualización por medio de instancias de ruteo (routing-instances)



1. INSTANCIAS DE RUTEO USUARIO-DEFINIDAS. De tal forma que se tenga una mayor flexibilidad, dentro de los equipos Juniper existen distintos tipos de instancias de ruteo que pueden ser definidas por el administrador, de tal forma que las instancias puedan adaptarse a distintos escenarios. A continuación se detalla el listado.

- Forwarding: Utilizada para el envío de datos basado en filtros.
- L2VPN: Empleada para aplicaicones L2VPN. Se excluye MPLS VPLS, ya que existe un tipo de instancia dedicado.
- No-Forwarding: Su fin es separar redes grandes en entidades administrativas pequeñas.
- Virtual-router: Empleada para aplicaciones no relacionadas a conectividad VPN, como virtualización del equipo.
- Vpls: Utilizada para implementaciones LAN punto-multipunto remotas entre distintos sitios dentro de una VPN.

Figura#35: Instancia de ruteo VPLS e interfaz asociada



```
{master}
jrobles@[redacted]> show configuration routing-instances VPLS-INSTANCIA
description "Instancia VPLS de prueba";
instance-type vpls;
vlan-id 1133;
interface ge-1/0/2.1133;
route-distinguisher [redacted];
vrf-target target:[redacted];
protocols {
    vpls {
        site-range 20;
        no-tunnel-services;
        site ce-1 {
            site-identifier 1;
        }
    }
}

{master}
jrobles@[redacted]> show configuration interfaces ge-1/0/2.1133
description "Interfaz dentro de una isntancia VPLS";
encapsulation vlan-vpls;
vlan-id 1133;
family vpls;
```

- Vrf (Virtual routing and forwarding): Su uso es para implementaciones L3VPN. Se muestra en la figura un ejemplo de este tipo y la configuración de la interfaz asociada.

Figura#36: Instancia de ruteo VRF e interfaces asociadas

```
{master}
jrobles@[redacted]01> show configuration routing-instances Instancia-VRF
instance-type vrf;
interface ge-1/1/1.1250;
route-distinguisher [redacted];
vrf-target target:[redacted];
vrf-table-label;
routing-options {
    static {
        route 192.168.11.0/24 next-hop 10.20.30.2;
    }
}

{master}
jrobles@[redacted]> show configuration interfaces ge-1/1/1.1250
description "Configuracion de interfaz dentro de una instancia VRF";
vlan-id 1250;
family inet {
    address 10.20.30.1/29;
```

2. SOBRE *ROUTE-DISTINGUISHER* Y *VRF-TARGET.* Resulta necesario notar la diferencia entre estos dos parámetros utilizados dentro de las instancias de ruteo que especifican conectividad por medio de una implementación VPN. Route-Distinguisher es utilizado para especificar una dirección única dentro de la red MPLS. Está conformado por dos bloques, la IP de conectividad del PE más un identificador único de la ruta. Esto conforma la denominada dirección VPNv4. De esta forma estos equipos pueden identificar de donde proviene un paquete y a que instancia de ruteo pertenece, haciéndolo único dentro de la red MPLS. VRF-Target determina qué prefijos pueden importarse y exportarse del PE en función de la instancia de ruteo, desde la tabla de rutas del equipo. Ambos parámetros realizan funciones distintas, pero en conjunto aseguran la virtualización de los servicios.

I. HERRAMIENTAS DE MONITOREO

En este apartado del marco teórico, se detalla la teoría necesaria que debe ser comprendida sobre las herramientas de monitoreo utilizadas, tanto para servicios de capa de red como para servicios de capa de enlace de datos.

1. RPM (REAL-TIME PERFORMANCE MONITORING). Es una propiedad de los equipos Juniper que les permite a los operadores de red una medición asertiva y constante del desempeño entre dos puntos de una red. Con esta herramienta, es posible configurar y enviar sondas (probes) hacia un objetivo (target) específico con el fin de monitorear su comportamiento y determinar factores cuyos valores puedan afectar la conexión hacia el mismo. Estos factores pueden ser pérdidas de paquetes, el denominado Round-Trip-Time (RTT), y jitter. De esta forma es posible especificar numéricamente el desempeño de la red, ofreciendo un monitoreo de nivel. Estos valores son determinados por solicitudes GET de HTTP, ICMP, TCP y UDP, dependiendo de la configuración. Para fines de este trabajo, interesan únicamente las solicitudes ICMP, específicamente ICMP-ping (a través de direcciones IPv4), ya que únicamente interesa garantizar la conectividad de un servicio punto a punto y normalmente los equipos destino son propiedad del suscriptor. En forma simple, RPM puede verse como tener un cliente y un servidor, que corresponden a dos puntos (usualmente remotos) de una red que desea monitorearse. El cliente envía periódicamente y de forma automatizada consultas específicas al servidor, el cual responde. Las respuestas del servidor determinan el estado de la red entre los dos puntos.

A continuación se detallan los componentes de una configuración RPM.

- Sonda de RPM (RPM Probe): Permite obtener estadísticas de desempeño con respecto a un destino de la sonda (probe target) identificado por medio de una dirección IP o una dirección URL. Cuando el objetivo recibe la sonda, el mismo responde al dispositivo que la genera. Determinando el comportamiento de factores mencionados como el tiempo de respuesta, el dispositivo que genera la sonda puede determinar estadísticas de desempeño. A continuación, se listan los distintos tipos de sonda. Como se mencionó, para interés de este trabajo, el tipo ICMP-ping es el único trascendente.

    - Solicitudes GET de HTTP hacia un destino URL.
    - Solicitudes GET para metadata hacia un destino URL.
    - Solicitudes ICMP-ping hacia una dirección destino (por defecto).
    - Solicitudes timestamp de ICMP hacia una dirección destino.
    - Paquetes ping de UDP hacia un dispositivo destino.
    - Solicitudes de timestamp de UPD hacia una dirección destino.
    - Paquetes ping de TCP hacia un dispositivo destino.

  Cabe destacar que las sondas de tipo UDP y TCP requieren que el destino sea configurado como un receptor de RPM, tal que pueda generar respuesta de las sondas. Esto puede no resultar escalable desde el punto de vista del suscriptor en el caso de no gestionar los dispositivos destino.

- Prueba de RPM (RPM Test): Cada destino de sonda es monitoreado a partir de una prueba de RPM. Una prueba de RPM forma parte de la sonda. En realidad, una sonda puede tener varias pruebas RPM, una por destino. Sin embargo se deben considerar dos definiciones

distintas para el término *Probe.* La definición de sonda, y la definición como un conjunto de pruebas individuales, por ejemplo un conjunto de pruebas de ping. En ese sentido una prueba de RPM (*test*) suele contener un conjunto de *probes* (pruebas de ping) y una sonda (*Probe)* contiene un conjunto de pruebas RPM (*Tests).* De estas pruebas RPM puede obtenerse estadísticas como desviación estándar y jitter.

- Intervalos de Pruebas RPM (Test) y de pruebas individuales (probes): Dentro de una prueba de RPM, las pruebas individuales son enviadas a intervalos de tiempo determinados y configurables en segundos. Cuando el número total de pruebas individuales es enviado dentro de los intervalos de tiempo especificados, una prueba RPM (test) se completa. El inicio de la siguiente prueba RPM (test), depende del intervalo de tiempo configurado en segundo entra cada prueba RPM (test).

- Cliente RPM: Se refiere al dispositivo que realiza la solicitud de RPM.

- Servidor RPM: Es el dispositivo al cual se le hace la solicitud.

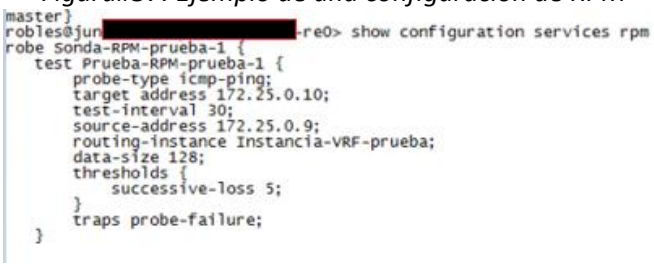
- Estadísticas de RPM: Al final de cada prueba RPM, el dispositivo colecciona estadísticas que permiten obtener el monitoreo de nivel. Se detallan las más importantes.

    - RTT mínimo: El menor tiempo medido durante una prueba RPM.
    - RTT máximo: El máximo tiempo medido durante una prueba RPM.
    - RTT promedio: El promedio medido durante una prueba RPM.
    - RTT desviación estándar: medida durante una prueba RPM.
    - Jitter: Diferencia entre el RTT mínimo y máximo medidos durante una prueba RPM.
    - Conteo de pruebas individuales (ping): Parámetro que especifica la cantidad de pruebas de ping durante una prueba RPM.
    - Porcentaje de pérdida: Es el porcentaje de pérdida de paquetes en relación a la cantidad de pruebas de ping enviados para los cuales no se tuvo respuesta.

- Umbral y trampas RPM: Es posible configurar umbrales para los tiempos de RTT para cada prueba de ping, así como para la desviación estándar y jitter medidos para cada prueba de RPM. Adicionalmente se puede configurar umbrales para pruebas de ping perdidas consecutivamente dentro de una prueba RPM y para el total de dichas pruebas dentro de una prueba RPM. Si el resultado dentro de una prueba RPM excede alguno de los umbrales configurados, el dispositivo puede generar un mensaje *syslog* y envía notificaciones o trampas vía SNMP. Tanto el mensaje syslog como las trampas SNMP deben haber sido previamente configuradas.

a. Configuración de RPM. En la figura a continuación, se observa el ejemplo de una configuración RPM en un equipo Juniper. Puede observarse el nombre de la sonda y que en este caso únicamente contiene una prueba RPM (test). La configuración indica que la prueba RPM es del tipo icmp-ping, (las pruebas individuales son pruebas de ping), la dirección destino, la instancia de ruteo (la fuente), la dirección origen (en este caso para especificar una subinterfaz específica de la instancia), el tamaño de cada paquete (MTU), el umbral que especifica 5 pérdidas de ping consecutivas, y la trampa generada en consecuencia de alcanzarse dicho umbral. Estos son algunos de los parámetros posibles a configurar dentro de una prueba RPM. Los no especificados son tratados por el sistema operativo del equipo con valores por defecto. En breve, se explica la función de cada parámetro.

Figura#37: Ejemplo de una configuración de RPM



```
master}
robles@jun[redacted]-re0> show configuration services rpm
robe Sonda-RPM-prueba-1 {
   test Prueba-RPM-prueba-1 {
       probe-type icmp-ping;
       target address 172.25.0.10;
       test-interval 30;
       source-address 172.25.0.9;
       routing-instance Instancia-VRF-prueba;
       data-size 128;
       thresholds {
           successive-loss 5;
       }
       traps probe-failure;
   }
```

Es importante destacar que en estos equipos la configuración es jerárquica. Como se explicó anteriormente, una prueba RPM (test) está contenida dentro de una sonda RPM (Probe). Una sonda puede contener varias de estas pruebas. Es necesario al configurar una sonda proceder a realizar la configuración de al menos una prueba RPM. Dentro de los parámetros más importantes para los objetivos de este trabajo destacan los siguientes:

- *Probe-type:* Tipo de Sonda RPM. El utilizado dentro de este trabajo es el ICMP-Ping, el cual se basa en realizar pruebas de ping hacia el destino para determinar el desempeño de la conexión. Si no se especifica, por defecto se escoge precisamente este tipo.

- *Probe-count:* Es la cantidad de pruebas de ping lanzadas dentro de una prueba RPM (test). El rango válido es entre 1 y 15, siendo por defecto, 1.

- *Probe-interval:* Es la cantidad de tiempo (en segundos) entre cada prueba de ping dentro de una prueba RPM. El rango es entre 1 y 255 segundos, siendo 3, por defecto.

- *Test-interval:* Es la cantidad de tiempo (en segundos) entre cada prueba RPM (test). El rango es entre 1 y 86400 segundos, siendo por defecto 0. Es decir, que si no se especifica un valor, la sonda se detiene luego de la primera prueba de RPM.

- *Target-address:* Es la dirección IP destino hacia la cual se desea realizar el monitoreo en el caso de una sonda ICMP-Ping.

- *Source-address:* Es la dirección IP fuente desde la cual se realizará el monitoreo. Puede ser opcional si se especifica en su lugar una instancia de ruteo con alguna subinterfaz configurada.

- *Routing-instance:* Se refiere a la instancia de ruteo desde la cual se realizará el monitoreo. Se escoge una subinterfaz dentro de ella a partir de la cual realizar las pruebas. Si una instancia posee más de una subinterfaz configurada, puede especificarse el parámetro *source-address* para forzar a realizar el monitoreo desde una de ellas.

- *Data-size:* Se refiere al tamaño del paquete (MTU) de cada prueba de ping. Se pueden especificar valores desde 0 hasta 65507. El default es 0.

- *History-size:* Especifica la cantidad de resultados de pruebas de ping almacenados en el registro histórico de la sonda. Valores de 0 a 255, siendo por defecto, 50.

- *Thresholds:* Umbral o valor sobre una medida de una prueba RPM que si es excedida genera un mensaje syslog o una trampa vía SNMP. Existen varios tipos de umbrales, de los que destacan:
  - *Jitter-rtt:* sobre el valor máximo de Jitter por prueba de RPM de 0 a 60,000,000 en microsegundos.
  - *Rtt:* sobre el valor máximo de RTT por prueba de ping dentro de una prueba RPM.
  - *Succesive-loss:* sobre la pérdida consecutiva de pruebas de ping dentro de una prueba de RPM. Por defecto, el valor es 1.
  - *Total-loss:* sobre la pérdida total de pruebas de ping dentro de una prueba RPM. Por defecto, el valor es 1.

- *Traps:* Notificaciones que son enviadas sin un umbral es alcanzado o excedido. Destacan las siguientes:
  - *Jitter-exceeded:* se activa al detectar que el Jitter iguala o excede el valor configurado dentro de una prueba de RPM.
  - *Probe-failure:* se activa cuando se iguala o supera el umbral de pérdidas de ping consecutivas.
  - *Rtt-exceeded:* genera notificaciones cuando se supera o iguala el valor máximo de RTT.
  - *Test-failure:* genera notificaciones cuando se iguala o supera la pérdida total de pruebas de ping por prueba de RPM.

b. Monitoreo de sondas RPM. Desde el equipo configurado, es posible determinar resultados recientes e históricos acerca de las estadísticas generadas por el monitoreo. Sin embargo, la configuración de archivos Syslog que generen mensajes con la fecha histórica de ciertos eventos identificados por RPM, resulta crucial en establecer el monitoreo de nivel. La generación de trampas vía SNMP puede resultar también conveniente. Dentro de los objetivos de este trabajo, está la generación de registros o logs que se almacenen de forma temporal en equipos y permanente en servidores especializados para que el operador pueda elaborar un mejor diagnóstico de incidentes. Debido a ello, el enfoque en este marco teórico obedece al uso de la herramienta Syslog. A continuación, se detalla los dos tipos de resultados que refleja el monitoreo RPM como tal.

- Resultados históricos: despliega información estándar acerca de las últimas 50 pruebas de ping por cada prueba de RPM (por defecto) dentro de la sonda. El historial se puede modificar desde 0 hasta 255. Como puede verse en la figura, la información que se despliega se puede dividir en lo siguiente: *Owner* (sonda), *test* (prueba RPM), *Probe received* (fecha a la que se recibe respuesta del ping) y *Round Trip time* (tiempo de ida y respuesta de una prueba de ping).

Figura#38: Despliegue de resultados históricos de una sonda RPM

```
user@host> show services rpm history-results
     Owner, Test                 Probe received              Round trip time
    p1, t1                      Wed Aug 12 01:02:35 2009             315 usec
    p1, t1                      Wed Aug 12 01:02:36 2009             266 usec
    p1, t1                      Wed Aug 12 01:02:37 2009             314 usec
    p1, t1                      Wed Aug 12 01:02:38 2009             388 usec
    p1, t1                      Wed Aug 12 01:02:39 2009             316 usec
    p1, t1                      Wed Aug 12 01:02:40 2009             271 usec
    p1, t1                      Wed Aug 12 01:02:41 2009             314 usec
    p1, t1                      Wed Aug 12 01:02:42 2009            1180 usec
```

- Resultados de sonda RPM: Los resultados de sonda ofrecen un registro verdaderamente histórico al almacenar un resumen de resultados globales, a diferencia del despliegue de los Resultados históricos, que únicamente ofrecen como máximo las últimas 255 pruebas de ping. Sin embargo, los Resultados de sonda son menos específicos. Los resultados de sonda se muestran por sonda y por prueba de RPM. Muestran información general como *Target address, Probe type, Test size, y Routing instance.* Los resultados de sonda se dividen en tres partes:

  - Resultados sobre la prueba RPM actual.
  - Resultados sobre la última prueba RPM
  - Resultados globales de prueba RPM.

  Por cada parte, se despliega la siguiente información:

  - Pruebas de ping enviadas.
  - Pruebas de ping recibidas.
  - Porcentaje de error.
  - RTT mínimo.
  - RTT máximo.
  - RTT promedio.
  - Jitter.
  - Desviación estándar.

  En la siguiente figura, se muestra el despliegue de información por sonda de RPM.

Figura#39: Despliegue de resultados de sonda RPM

```
user@host> show services rpm probe-results
Owner: Rpm-Bgp-Owner, Test: Rpm-Bgp-Test-1
    Target address: 10.209.152.37, Probe type: icmp-ping, Test size: 5 probes
    Routing Instance Name: LS1/RI1
    Probe results:
      Response received, Fri Oct 28 05:20:23 2005
      Rtt: 662 usec
    Results over current test:
      Probes sent: 5, Probes received: 5, Loss percentage: 0
      Measurement: Round trip time
        Minimum: 529 usec, Maximum: 662 usec, Average: 585 usec,
        Jitter: 133 usec, Stddev: 53 usec
    Results over all tests:
      Probes sent: 5, Probes received: 5, Loss percentage: 0
      Measurement: Round trip time
        Minimum: 529 usec, Maximum: 662 usec, Average: 585 usec,
        Jitter: 133 usec, Stddev: 53 usec
```

2. SYSLOG (SYSTEM LOGGING) EN JUNIPER. Estándar para registro de eventos que ocurren dentro de un sistema operativo de un equipo en una red IP. Definido dentro del RFC 5424, syslog permite la generación de mensajes que identifican de forma única el evento ocurrido, con un nombre, fecha y descripción del mismo. El sistema operativo de un equipo es capaz de generar mensajes que identifiquen falla o éxito en la conectividad dentro de la red. El monitoreo de sondas de RPM que ofrece el Junos OS puede resultar insuficiente al momento de revisar puntos críticos en la red, que requieren de información más detallada. Syslog, en equipos Juniper, ofrece esta posibilidad en conjunto con RPM, haciendo posible segmentar una red y así determinar el punto exacto del inconveniente presentado.

Específicamente, en dispositivos que corren el sistema operativo *Junos OS*, este mecanismo es utilizado para el registro de operaciones de alto nivel, como interfaces que cambian de estado (*Up/Down)*, o usuarios que entran o salen del equipo. El sistema operativo almacena los registros de las distintas operaciones ejecutadas en archivos dentro del directorio */var/log*. El archivo syslog principal y que por defecto incluye el Junos OS en todas las configuraciones de fábrica es el */var/log/messages*. El sistema operativo en equipos Juniper soporta distintos niveles de recursos y severidad. El nivel de recurso define la clase de registro syslog y el nivel de severidad, el nivel de detalle del mismo. La información generada por medio de syslog puede ser almacenada dentro del archivo */var/log/messages*, dentro de un archivo configurado o bien enviado a un servidor o host remoto. Ambas vías, registro por achivos y registro por servidor remoto, son recomendadas por el fabricante de los equipos para un mejor diagnóstico de los eventos presentados dentro de la red.

a. Partes de un mensaje syslog. Tal y como se define en el RFC 3164, el formato completo de un mensaje syslog se compone de tres partes discernibles. La primera parte PRI, la segunda el encabezado (header) y la tercera, el mensaje. El paquete debe contener en total 1024 bytes o menos. No existe un mínimo de bytes, pero el envío de un mensaje syslog sin información no tiene valor y no debería ser empleado.

- PRI: es el valor de prioridad que se le da al mensaje syslog. Este valor se encuentra contenido dentro de los caracteres “<” y “>”, y se compone de entre 3 y 5 caracteres ASCII, incluyendo los signos menor y mayor. El valor de prioridad se compone del nivel de recurso y de severidad del mensaje. Estos niveles son numéricamente codificados con valores decimales. El cálculo del valor de prioridad se realiza multiplicando primeramente el valor de nivel de recurso por ocho y luego sumando el valor de nivel de severidad. Así, por ejemplo, un mensaje kernel (Recurso=0) con una severidad de emergencia (Severidad=0), tendrá una prioridad de 0. Por otro lado, un mensaje “Uso local 4” (Recurso=20) con una severidad de aviso (*Notice*) (Severidad=5), tendrá una prioridad de 165. Como puede verse, mientras menor sea el valor de prioridad, mayor es la importancia que le da el sistema operativo del equipo.

  A continuación en las siguientes figuras, se muestra la designación de los niveles de recurso y severidad, respectivamente, así como los códigos asociados.

Figura#40: Niveles de recurso en syslog

Numerical Code	Facility
0	kernel messages
1	user-level messages
2	mail system
3	system daemons
4	security/authorization messages
5	messages generated internally
6	line printer subsystem
7	network news subsystem
8	UUCP subsystem
9	clock daemon (note 2)
10	security/authorization message
11	FTP daemon
12	NTP subsystem
13	log audit (note 1)
14	log alert (note 1)
15	clock daemon (note 2)
16	local use 0 (local0)
17	local use 1 (local1)
18	local use 2 (local2)
19	local use 3 (local3)
20	local use 4 (local4)
21	local use 5 (local5)
22	local use 6 (local6)
23	local use 7 (local7)

Figura#41: Niveles de severidad en syslog

Numerical Code	Severity
0	Emergency: system is unusable
1	Alert: action must be taken immediately
2	Critical: critical conditions
3	Error: error conditions
4	Warning: warning conditions
5	Notice: normal but significant condition
6	Informational: informational messages
7	Debug: debug-level messages

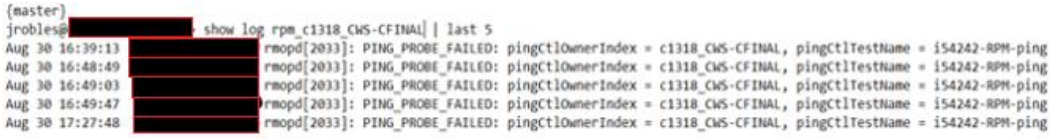
- Encabezado syslog: se compone de dos campos únicamente, la marca de tiempo (*timestamp*) y el nombre del host o dirección IP del dispositivo. La marca de tiempo sigue luego del carácter “>” utilizado en la parte PRI y un caracter de espacio solamente es utilizado luego de cada uno de estos campos. La marca de tiempo, como su nombre lo indica, permite segmentar en el tiempo la ocurrencia del evento. La misma se representa en el formato “*Mmm dd hh:mm:ss*”. El nombre del host, de no estar definido dentro de la configuración del equipo es sustituido por su dirección IP. Si un equipo posee múltiples direcciones IP, escoge la de la interfaz de salida. Existe una alternativa de fijar una única dirección IP para esto, tal que pueda representar un identificador del equipo.

- Mensaje syslog: complemento del paquete syslog. Contiene el nombre del proceso que lo genera, el código del mensaje y el texto del mensaje. El primero identifica o nombra al proceso que genera el mensaje. El código de mensaje identifica el evento generado. Es importante conocer los códigos que se refieran al evento que se busca registrar, ya que van de la mano con los umbrales generados en la configuración de RPM. Por su lado, el texto del mensaje brinda información detallada del evento.

A continuación, se muestra un ejemplo de un registro generado que identifica unívocamente un evento ocurrido en la red. Este evento “*PING_PROBE_FAILED*” ocurre cuando el umbral de

"*successive-loss"* (pruebas de ping perdidas de forma consecutiva) se ha igualado o superado. Es importante recordar que por defecto, el valor de este umbral es de 1 si no se especifica dentro de la jerarquía de la configuración.

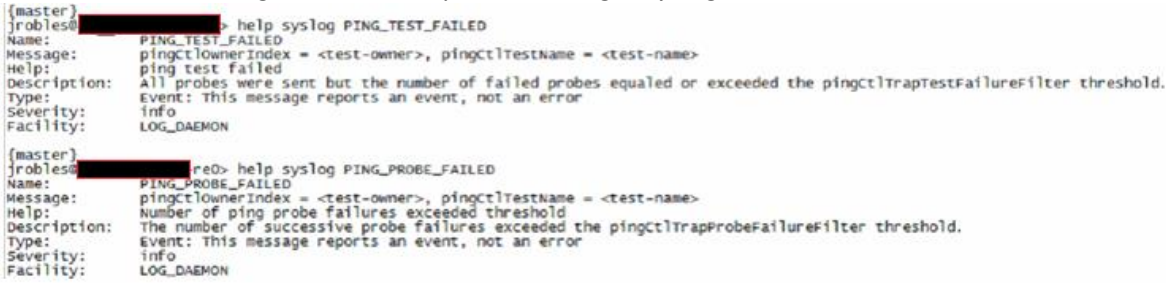
Figura#42: Registros dentro de un archivo syslog



Como se mencionó con anterioridad, es importante conocer los códigos que se refieran a eventos syslog deseados; en este caso, eventos que se refieran a fallas en las pruebas de ping dentro de una prueba RPM y que vayan ligados a umbrales dentro de una configuración RPM.

La siguiente figura muestra la relación: *"Successive-loss"*- "*PING_PROBE_FAILED"* y "*Total-loss"-"PING_TEST_FAILED".*

Figura#43: Descripción de códigos syslog de interés



b. Configuración syslog. En la siguiente figura se muestra la configuración jerárquica de syslog dentro de un sistema operativo Junos. Se detallan, a continuación, los aspectos más importantes.

Figura#44: Configuración syslog en Juniper

```
syslog {
    archive {
        (binary-data| no-binary-data);
        files number;
        size maximum-file-size;
        start-time "YYYY-MM-DD.hh:mm";
        transfer-interval minutes;
        (world-readable | no-world-readable);
    }
    console {
        facility severity;
    }
    file filename {
        facility severity;
        explicit-priority;
        match "regular-expression";
        archive {
            (binary-data| no-binary-data);
            files number;
            size maximum-file-size;
            start-time "YYYY-MM-DD.hh:mm";
            transfer-interval minutes;
            (world-readable | no-world-readable);
        }
        structured-data {
            brief;
        }
    }
```

```
    host (hostname | other-routing-engine | scc-master) {
        facility severity;
        explicit-priority;
        facility-override facility;
        log-prefix string;
        match "regular-expression";
        source-address source-address;
        structured-data {
            brief;
        }
        port port number;
    }
    log-rotate-frequency frequency;
    source-address source-address;
    time-format (millisecond | year | year millisecond);
    user (username | *) {
        facility severity;
        match "regular-expression";
    }
}
```

- *Archive:* Configura propiedades de registro para todos los registros syslog si está en el nivel de configuración *[edit system syslog].* Se compone de los siguientes parámetros:

    - *files number:* El número de registros de los que se compone el archivo syslog. Cuando un archivo del registro ha llegado al tamaño máximo de bytes especificado, genera un registro para almacenar la información antigua y seguir registrando nuevos eventos. Cada nuevo registro se va generando conforme sea necesario. Esto lo realiza, identificándolos con la siguiente nomenclatura: *logfile, logfile.0.gz, logfile.1.gz, logfile.2.gz… logfile.n.gz*, donde "*n*" representa el número de registros de los que se compone el archivo. *Logfile* es el nombre del archivo. Luego de esto, los registros más antiguos se van perdiendo. El rango es de 1 a 1000 y por defecto, 10.

    - *size:* especifica el tamaño de cada registro del archivo syslog. Es el tamaño máximo que el registro original soporta para generar luego el *logfile.0.gz* y así sucesivamente conforme resulte necesario. Rango de 64KB a 1GB, y por defecto, 1MB en routers Juniper de las series MX.

    - *binary-data | no-binary-data:* marca el archivo como uno de datos binarios. Esto permite el correcto registro de este tipo de datos. Por defecto, *no-binary-data.*

    - *world-readable | no-word-readable:* permite a todos los usuarios ver el archivo syslog o únicamente al usuario root y a los usuarios con el permiso denominado "*Maintenance"*. Es importante mencionar que por defecto se utiliza *no-world-readable.*

- *File filename:* Especifica parámetros de syslog para un archivo en específico, el cual es identificado por el nombre configurado (*filename*). Los parámetros no especificados dentro de esta jerarquía son descritos individualmente en su propio apartado.
    - *facility severity:* se refiere al nivel de recurso y severidad que se le da al archivo.
    - *explicit priority:* especifica el valor de prioridad explícitamente sin especificar por separado los valores de recurso y severidad.
    - *structured-data brief:* registra los mensajes syslog en un format específico IETF. Esta opción, al ser especificada, hace que el Junos OS ignore la configuración de *explicit priority* y de *time-format.*

- *Match regular-expression:* Se detalla un *string* o cadena de caracteres que debe aparecer en el mensaje para ser registrado dentro de un archivo o servidor syslog. Es de vital importancia para delimitar los mensajes almacenados dentro del archivo o enviados al servidor y es útil para especificar códigos syslog deseados. Esta configuración entra dentro de la jerarquía *"File filename" y "Host hostname".*

- *Host hostname:* Configura la generación de mensajes syslog para ser enviadas a un destino remoto, en este caso, un servidor syslog. Los parámetros dentro de la jerarquía no explicados en este apartado son definidos individualmente, en su propio apartado.
  - *log-prefix:* especifica un *string* o cadena de caracteres como parte del mensaje enviado al servidor syslog. De utilidad para identificar el mensaje de forma apropiada.
  - *port port-number:* especifica el puerto destino para el servidor syslog. Rango entre 0 y 65535, donde por defecto se utiliza el puerto 514.

- *Log-rotate-frequency:* Frecuencia de tiempo en la que el Junos OS revisa el tamaño de los archivos syslog para determinar la creación de nuevos registros o el descarte de mensajes antiguos. Rango entre 1 y 59 minutos, siendo por defecto, 15.

- *Source-address:* Dirección IP dentro del equipo cliente a partir de la cual se hará la solicitud a los servidores remotos. Si esta configuración se encuentra dentro de la jerarquía de un host específico, únicamente aplica al mismo.

- *Time-format:* Especifica la información a incluir en el formato de la marca de tiempo.

- *User (username | *):* Determina el envío de mensajes syslog a usuarios determinados reconocidos por el equipo.
  - ** (asterisco):* determina que el envío de mensajes se hará a todos los usuarios.
  - *username:* envío de mensajes a un usuario particular. Para incluir más usuarios, simplemente se deben especificar dentro de la misma línea de configuración.

En la siguiente figura, se muestra un ejemplo simple de la configuración de dentro de la jerarquía syslog de un equipo Juniper.

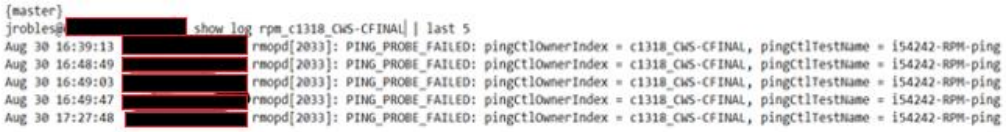
Figura#45: Ejemplo de configuración syslog en Juniper

```
{master}
jrobles@[redacted]-re0> show configuration system syslog
user * {
    any emergency;
}
host [redacted].24.107 {
    any info;
    match "PING_TEST_FAILED|PING_PROBE_FAILED";
    log-prefix Juniper[redacted];
    source-address [redacted].1.31;
}
file rpm_c1318-CWS-CFINAL {
    any info;
    match "PING_PROBE_FAILED: pingCtlOwnerIndex = c1318-CWS-CFINAL";
    archive size 2m files 2 world-readable;
}
```

c. Monitoreo syslog. El sistema operativo de un equipo Juniper permite ejecutar comandos de operación para la revisión de mensajes syslog, haciendo referencia a uno, varios o todos los archivos syslog o de ingreso y salida de usuarios. Así mismo, permite hacer un reinicio de un archivo específico.

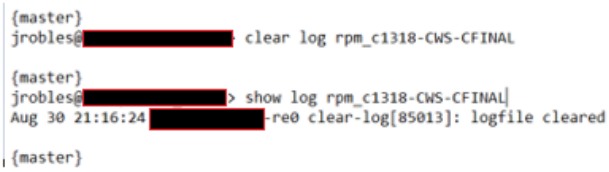
En la siguiente figura se muestra cómo revisar eventos dentro de un archivo syslog específico.

Figura#46: Revisión de un archivo syslog



En la siguiente figura se muestra cómo hacer un reinicio de un archivo específico.

Figura#47: Reinicio de archivo syslog



3. NETCONF Y JUNOS XML API. NETCONF es un protocolo de gestión basado en XML utilizado para manejar la comunicación entre aplicaciones programadas y dispositivos con sistema operativo Junos. Las aplicaciones programadas son denominadas aplicaciones cliente, y son establecidas a través de herramientas basadas en lenguajes de programación (como Junos PyEZ). Dichas aplicaciones solicitan y cambian la configuración de los equipos de red utilizando NETCONF como enlace de comunicación con Junos OS. NETCONF es interno al dispositivo con Junos OS y ejecuta la función de servidor. NETCONF utiliza una codificación de datos basada en XML para monitoreo y configuración de equipos por medio de RPCs (Remote Procedure Call). Las aplicaciones utilizan las operaciones del protocolo para visualizar, editar, y declarar cambios en la configuración de los equipos, tal y como los administradores de red utilizan los comandos de operación y configuración para realizar estas tareas.

Junos XML API, por su parte, es una representación en formato XML de los comandos de operación y enunciados de configuración en Junos OS. Esta información se representa en formato ASCII y es así como se presenta al usuario. Junos XML API define un equivalente para cada comando de configuración y operación a través de elementos tag de solicitud, así como sus respectivas salidas, a través de elementos tag de respuesta. Un RPC es la representación utilizada por aplicaciones cliente para hacer referencia a un elemento tag de solicitud. Junos XML API puede dividirse en jerarquías de elementos tag. Cuando una aplicación cliente modifica un dispositivo con Junos OS, los elementos tag de configuración son los utilizados por NETCONF. Los mismos son equivalentes a los comandos de configuración tradicionales. Por otro lado, los elementos de tag de operación son equivalentes a los comandos usados normalmente para obtener información del equipo.

El protocolo de gestión de NETCONF es descrito en el RFC 4741. Las aplicaciones cliente solicitan o cambian la información en equipo de red mediante la codificación de la solicitud por medio de los elementos tag del protocolo, y luego la envían a la aplicación servidor de NETCONF en el dispositivo

involucrado. En los dispositivos con sistema operativo Junos, el servidor NETCONF está integrado en el Junos OS. El servidor NETCONF dirige la solicitud a los módulos de software apropiados dentro del dispositivo, codifica la respuesta por medio de los elementos tag de NETCONF, y devuelve el resultado a la aplicación cliente. Por ejemplo, para solicitar información acerca del estado de las interfaces de un equipo, la aplicación cliente envía un RPC con el elemento tag *<get-interface-information>* del Junos XML API. La aplicación servidor de NETCONF obtiene la información, presentándola dentro del elemento tag *<interface-information>*.

a. Ventajas del uso de NETCONF y de Junos XML API

- Documentación completa de todas las opciones de configuración y de operación disponible dentro del Junos OS.
- Los nombres de los elementos tag indican claramente la función de un elemento en una solicitud operacional o declaración de configuración.
- La combinación de nombres significativos de elementos tag y las reglas estructurales en un DTD (*Document Type Definition,* se define más adelante), hacen que sea fácil comprender el contenido y la estructura de una serie de RPCs.
- El uso de elementos de tag hace que el análisis o "parseo" de la información sea mucho más sencillo y escalable.
- Es más fácil visualizar diferentes detalles sobre un componente del dispositivo según sea requerido.

En la siguiente figura se muestra el despliegue de cierta información de un dispositivo Junos OS en su formato tradicional ASCII, y su versión XML etiquetada (tagged).

Figura#48: Comparación formato ASCII y XML en Netconf

```
Physical interface: fxp0, Enabled, Physical link is Up
    Interface index: 4, SNMP ifIndex: 3

The corresponding XML-tagged version is:

<interface>
    <name>fxp0</name>
    <admin-status>enabled</admin-status>
    <operational-status>up</operational-status>
    <index>4</index>
    <snmp-index>3</snmp-index>
</interface>
```

Cuando una aplicación cliente necesita extraer un valor específico de la salida de formato ASCII, debe depender de la ubicación del valor. Por ejemplo, la aplicación cliente quiere extraer el índice de interfaz (*interface index)*. Para ello, puede utilizar una serie de búsquedas de expresiones regulares para localizar cadenas específicas, pero una de las dificultades es que el número de dígitos en el índice de la interfaz no es necesariamente predecible. La aplicación cliente no puede simplemente leer un número determinado de caracteres después de la etiqueta *"interface index"*, sino que en su lugar debe extraer todo entre dicha etiqueta y la etiqueta posterior, que es *"SNMP ifIndex"*. Surge un problema con este algoritmo si el formato o el ordenamiento de salida cambian en una versión posterior del sistema operativo de Junos, ya que extraería información diferente a la esperada. La aplicación que extrae el número de índice de interfaz delimitada por las etiquetas *"interface index"* y *"SNMP ifIndex"* obtendría un resultado incorrecto.

En contraste, la naturaleza estructurada los elementos tag permite a una aplicación cliente recuperar el índice de interfaz mediante la extracción de todo dentro de la etiqueta de apertura *<index>* y la etiqueta de cierre *</index>.* La aplicación *no* tiene que depender de la posición de un elemento en la cadena de salida en formato ASCII ni de la versión del Junos OS, por lo que el servidor de NETCONF puede emitir los elementos dentro de la etiqueta sin problemas. La adición de un nuevo elemento, *"logical-index"*, por ejemplo, en una versión futura no afecta la capacidad de una aplicación para localizar el elemento de interés, dentro de la etiqueta *<index>*, y extraer su contenido.

Cuando un dispositivo utiliza para el análisis (parseo), un formato de salida ASCII, hay que diseñar y escribir rutinas especiales y estructuras de datos en el programa de la aplicación cliente, para extraer y almacenar la información necesaria para un nivel de detalle dado. XML hace que sea fácil de usar la misma rutina de extracción para varios niveles de detalle, simplemente haciendo caso omiso de los elementos de la etiqueta que no son necesarios y enfocándose en el de interés.

b. Descripción general de una sesión NETCONF. La comunicación entre el servidor NETCONF y una aplicación cliente se basa en el establecimiento de una sesión. Las dos partes establecen explícitamente una conexión antes de intercambiar datos y cierran la conexión al finalizar. Cada solicitud de la aplicación cliente y cada respuesta del servidor NETCONF constituyen un documento XML bien formado (well-formed), debido a que los flujos de etiquetas obedecen las reglas estructurales definidas en los DTDs, para el tipo de información que codifican. Es importante mencionar que las aplicaciones cliente acceden al servidor Netconf utilizando el protocolo SSH y su método estándar de autenticación. Luego, se determina por medio del usuario utilizado si es factible realizar las solicitudes de parte de la aplicación cliente.

A continuación, se detalla la estructura básica de una sesión NETCONF:

- La aplicación cliente establece una conexión con el servidor NETCONF y abre la sesión NETCONF vía SSH.
- El servidor de NETCONF y la aplicación cliente intercambian información de inicialización, la cual es usada para determinar si están utilizado versiones compatibles del sistema operativo de Junos y de NETCONF.
- La aplicación cliente envía una o más peticiones al servidor NETCONF y analiza (parsea) las respuestas recibidas del mismo, para obtener los valores deseados de forma íntegra.
- La aplicación de cliente cierra la sesión NETCONF y la conexión con el servidor NETCONF.

c. XML y Junos OS. XML (Extensible Markup Language) es un estándar para representar y comunicar información. Es un metalenguaje para definir etiquetas personalizadas que se aplican a un conjunto de datos o documento para describir la función de los elementos individuales y codificar las relaciones jerárquicas entre ellos. El sistema operativo Junos respalda de forma nativa XML para el funcionamiento y la configuración de los dispositivos. La interfaz de línea de comandos (CLI) del sistema operativo Junos y la infraestructura de Junos OS se comunican usando XML. Cuando se emite un comando de operación a través de CLI, éste convierte el comando en el formato XML para su procesamiento. Luego, Junos OS devuelve el resultado en forma de un documento XML, que la línea de comandos convierte de nuevo en un formato legible (formato ASCII) para desplegarlo al usuario. Dentro de Junos XML API, cada comando de modo de operación se mapea a un elemento tag de solicitud, devolviendo un elemento tag de respuesta.

Para revisar la documentación XML de comandos de operación en Junos OS, el usuario debe emitir un comando deseado y canalizar (*"pipe"*) la salida por medio de dos subcomandos distintos. El subcomando *"display xml"* despliega la información en elementos tag de respuesta. El subcomando *"display xml rpc"* despliega la información en elementos tag de solicitud. En la siguiente figura se muestra un ejemplo de lo mencionado.

Figura#49: Elementos tag de solicitud y respuesta

```
user@host> show chassis alarms
No alarms currently active
user@host> show chassis alarms | display xml
<rpc-reply xmlns:junos="http://xml.juniper.net/junos/10.4R1/junos">
    <alarm-information xmlns="http://xml.juniper.net/junos/10.4R1/junos-alarm">
        <alarm-summary>
            <no-active-alarms/>
        </alarm-summary>
    </alarm-information>
    <cli>
        <banner></banner>
    </cli>
</rpc-reply>
user@host> show chassis alarms | display xml rpc
<rpc-reply xmlns:junos="http://xml.juniper.net/junos/10.4R1/junos">
    <rpc>
        <get-alarm-information>
        </get-alarm-information>
    </rpc>
    <cli>
        <banner></banner>
    </cli>
</rpc-reply>
```

d. Descripción general de XML. XML es un lenguaje para la definición de un conjunto de marcadores, llamados "tags" o etiquetas, que se aplican a un conjunto de datos o documento para describir la función de los elementos individuales y codificar las relaciones jerárquicas entre ellos.

e. Elementos tag de Netconf: Los elementos de un documento o conjunto de datos compatible con XML siempre se incluyen en las etiquetas de apertura y cierre pareadas, tal y como se muestra en la figura a continuación.

Figura#50: Uso de elementos tag para identificar información

```
<interface-state>enabled</interface-state>
<input-bytes>25378</input-bytes>
```

XML es más estricto que HTML en este sentido, que a veces usa etiquetas únicamente de apertura. El término "*tag*" o etiqueta estrictamente se refiere a un conjunto de tres partes: la etiqueta de apertura, el contenido y la etiqueta de cierre. El contenido puede ser una cadena de caracteres alfanuméricos como en los ejemplos anteriores, o puede ser en sí mismo un elemento de etiqueta contenedor, que contiene otras etiquetas dentro de su jerarquía. Si un elemento de etiqueta no contiene información, *<snmp-trap-flag/>* por ejemplo, equivale al conjunto *<snmp-trap-flag>* y *</snmp-trap-flag>*. Los elementos tag de NETCONF obedecen la convención XML que indica que el nombre de dicho elemento indica el tipo de información que encierra.

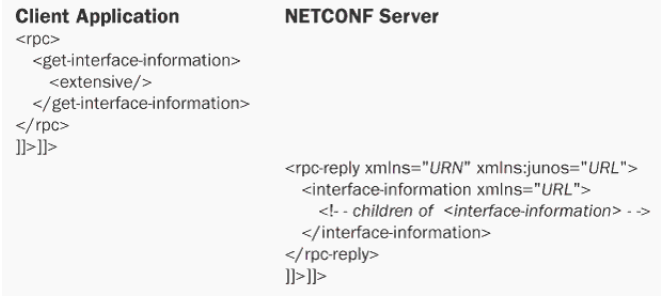
DTD (Document type definition): documento de definición que enumera todos los elementos de tag o de etiqueta que pueden aparecer dentro de una jerarquía de datos específica. Define las relaciones padre-hijo entre las etiquetas, y especifica otras características de las mismas. La misma DTD se puede aplicar a varios programas de aplicación cliente XML o conjuntos de datos.

f. Descripción general de las convenciones de NETCONF. Una aplicación cliente debe cumplir con las convenciones de XML y NETCONF para funcionar adecuadamente. Cada solicitud de la aplicación cliente debe ser un documento XML bien formado (well-formed); es decir, que debe obedecer las reglas estructurales definidas en los DTD para el tipo de información requerida en la solicitud. La aplicación cliente debe emitir los elementos de etiqueta en el orden requerido y sólo en los contextos legales, es decir, dentro de la jerarquía correcta. Del mismo modo, cada respuesta desde el servidor NETCONF constituye un documento XML bien formado (el servidor NETCONF obedece convenciones XML y NETCONF).

g. Elementos tag de solicitud y de respuesta: Un elemento tag de solicitud es generado por una aplicación cliente para solicitar información acerca de del estado actual o de la configuración de un dispositivo para modificarla. Un elemento tag de solicitud corresponde a un comando operacional o de configuración. Ocurre únicamente dentro de una etiqueta *<rpc>*. Un elemento de etiqueta de respuesta representa la respuesta del servidor de NETCONF a un elemento de solicitud y sólo se produce dentro de una etiqueta *<rpc-reply>*.

En la siguiente figura, se muestra la estructura de una etiqueta ejecutada por las aplicaciones cliente y procesada y respondida por el servidor.

Figura#51: Operaciones tag ejecutadas por aplicaciones cliente-servidor



h. Elementos de etiqueta secundarios de un elemento tag de solicitud: Algunos elementos de etiqueta de solicitud contienen elementos de etiqueta hijos o secundarios. Para solicitudes de configuración, cada elemento de etiqueta secundaria representa un elemento contenedor de configuración dentro de un nivel de jerarquía específico o bien, un objeto de configuración al final de la jerarquía. Para las solicitudes de operación, cada elemento etiqueta secundario representa una de las opciones que se ha suministrado en la línea de comandos cuando se emite el comando CLI equivalente. Algunas solicitudes tienen elementos de etiqueta secundarios obligatorios. Para hacer una solicitud con éxito, una aplicación cliente debe emitir los elementos de etiquetas obligatorias en las etiquetas de apertura y cierre del elemento etiqueta petición.

i. Elementos de etiqueta secundarios de un elemento tag de respuesta: Los elementos de etiqueta secundarios de un elemento de etiqueta de respuesta representan los elementos de datos individuales devueltos por el servidor de NETCONF para una solicitud particular. Los elementos

secundarios pueden ser tanto elementos individuales de etiquetas, como elementos de etiqueta contenedores que encierran sus propios elementos.

j. Espacios, Caracteres en nueva línea y otros espacios en blanco: Como dicta la especificación XML, el servidor NETCONF ignora los espacios en blanco (espacios, tabulaciones, caracteres de nueva línea, y otros caracteres que representan el espacio en blanco) que se producen entre los elementos tag generados por una aplicación cliente. Las aplicaciones cliente pueden, aunque no necesitan, incluir espacios en blanco entre los elementos tag. Sin embargo, no se deben insertar espacios en blanco dentro de una apertura o cierre de etiqueta. En sus respuestas, el servidor NETCONF incluye un espacio en blanco entre los elementos de etiqueta para mejorar la legibilidad de las respuestas que se guardan en un archivo. Utiliza caracteres de nueva línea para poner cada elemento de la etiqueta en su propia línea, y espacios para insertar sangrías en elementos tag secundarios hacia la derecha en comparación con sus elementos "padres". Una aplicación cliente puede ignorar o descartar el espacio en blanco, sin embargo, no debe depender de la presencia o ausencia de espacios en blanco en cualquier lugar en particular al analizar un flujo de etiquetas.

k. Comentarios XML: Las aplicaciones cliente y el servidor NETCONF pueden insertar comentarios XML en cualquier punto entre elementos tag, pero no dentro de ellos. Las aplicaciones cliente no deben depender del contenido referido a comentarios. Las aplicaciones cliente tampoco pueden usar los comentarios para transmitir información al servidor NETCONF, porque el servidor descarta automáticamente los comentarios que recibe. Los comentarios XML se incluyen dentro de las cadenas <*!*-- y -->,y no pueden contener la cadena *"--"*. El siguiente es un ejemplo de un comentario XML: *"<!-- This is a comment. Please ignore it. -->"*

l. Referencias de entidad predefinidas: Por convención de XML, hay dos contextos en los que caracteres determinados no pueden aparecer en su forma usual:

- En una cadena que aparece entre las etiquetas de apertura y cierre (el contenido del elemento de etiqueta).
- En el valor de cadena asignado a un atributo de una etiqueta de apertura.

Al incluir un carácter no permitido en cualquiera de estos dos contextos, las aplicaciones cliente deben sustituir dicho carácter por la referencia de entidad predefinida equivalente, que es una cadena de caracteres que representa el carácter no permitido. En la siguiente figura, se muestra las referencias predefinidas (derecha) para caracteres no permitidos (izquierda) entre etiquetas de apertura y cierre.

Figura#52: Referencias de entidad predefinidas

& (ampersand)	&amp;
> (greater-than sign)	&gt;
< (less-than sign)	&lt;

m. Mapeo de comandos de operación Junos OS a elementos tag XML. Como se ha mencionado, Junos XML API es una representación XML de los estados de configuración Junos OS y comandos del

modo de operación. Define un equivalente XML para todas las declaraciones en la jerarquía de configuración del sistema operativo Junos y muchos de los comandos emitidos en el modo de operación. Los elementos tag de solicitud son utilizados en las llamadas de procedimiento remoto (RPC) dentro de las sesiones cliente-servidor. Su fin es solicitar información a un dispositivo con sistema operativo Junos.

n. Mapeo de comandos con valores variables: La convención de XML requiere que la API XML Junos defina un elemento tag para cada opción posible dentro de un comando de operación. Así por ejemplo, como se muestra en la figura, el comando *"show interfaces"* puede especificar cualquier interfaz física o virtual, lo cual indica que es un comando con valores variables, y lo mismo ocurre con el comando *"show bgp neighbor"*, que puede especificar cualquier direccción IP.

Figura#53: Mapeo de comandos de operación con valores variables

CLI Command	JUNOS XML Tags
show interfaces t3-5/1/0:0	<rpc> <get-interface-information> <interface-name>t3-5/1/0:0</interface-name> </get-interface-information> </rpc>
show bgp neighbor 10.168.1.122	<rpc> <get-bgp-neighbor-information> <neighbor-address>10.168.1.122</neighbor-address> </get-bgp-neighbor-information> </rpc>

o. Mapeo comandos con valores fijos: Algunos comandos CLI incluyen opciones que tienen una forma fija, como las cadenas breves y las detalladas. La API XML Junos normalmente asigna esa opción para una etiqueta vacía cuyo nombre coincide con el nombre de la opción, tal y como se observa en la figura a continuación.

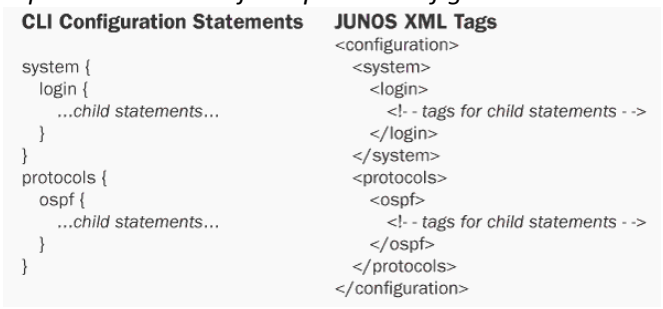
Figura#54: Mapeo de comandos de operación con valores fijos

CLI Command	JUNOS XML Tags
show isis adjacency detail	<rpc> <get-isis-adjacency-information> <detail/> </get-isis-adjacency-information> </rpc>

p. Mapeo de elementos de configuración Junos OS a elementos tag XML. La API XML Junos define un elemento tag para cada comando de configuración en la jerarquía. En los niveles superiores de la jerarquía de configuración, casi siempre hay una correspondencia uno-a-uno entre los elementos de tag y los comandos, y la mayoría de los nombres de etiqueta coincide con el nombre de sentencia configuración.

q. Mapeo de niveles de jerarquía de elementos contenedores: El elemento *<configuration>* es el elemento contenedor de nivel superior en Junos OS en el modo de configuración. Corresponde al nivel *[edit]* en dicho modo. La mayoría de los estados en los siguientes niveles de la jerarquía son contenedores. El elemento tag que corresponde a un elemento contenedor de configuración dado, casi siempre tiene el mismo nombre que dicho contenedor. En la siguiente figura, se muestra la correspondencia entre los niveles altos de la jerarquía de configuración Junos OS y los elementos tag XML asociados.

Figura#55: Mapeo de niveles de jerarquía de configuración a elementos tag XML



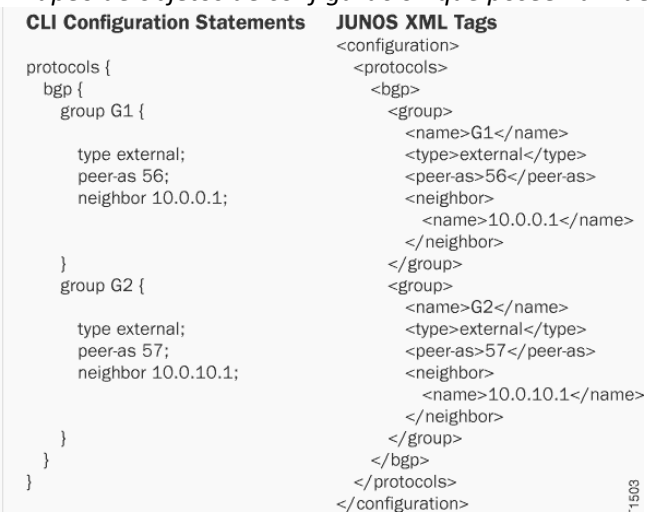
CLI Configuration Statements	JUNOS XML Tags
	<configuration>
system {	<system>
login {	<login>
...child statements...	<!- - tags for child statements - ->
}	</login>
}	</system>
protocols {	<protocols>
ospf {	<ospf>
...child statements...	<!- - tags for child statements - ->
}	</ospf>
}	</protocols>
	</configuration>

r. Mapeo de objetos que tienen un identificador: En algunos niveles de jerarquía, un mismo objeto puede aparecer varias veces. Cada instancia del objeto tiene un identificador único para distinguirlo de otros casos. En la notación de CLI, la declaración de los elementos principales o "padres" para tal objeto se compone de una palabra clave y el identificador de la siguiente forma:

keyword identifier {
… configuration statements for individual characteristics …
}

donde *"keyword*" es el tipo de objeto que está siendo definido, y "*identifier"* es el nombre único o identificador de dicha instancia. El identificador debe ser encerrado en un elemento tag propio. Es frecuente que este elemento se denomine *<name>*. La configuración para la mayoría de los objetos que tienen identificadores incluye declaraciones hijas adicionales, las cuales representan otras características del objeto. Por ejemplo, en la figura a continuación, cada grupo BGP configurado en el nivel *[edit protocols bgp group]* tiene un nombre asociado (identificador) y puede tener declaraciones hijas (leaf) adicionales como sistema autónomo (AS) y dirección IP del vecino. Estas se identifican con una etiqueta propia, es decir, cuyo nombre es igual al del *keyword.*

Figura#56: Mapeo de objetos de configuración que poseen un identificador



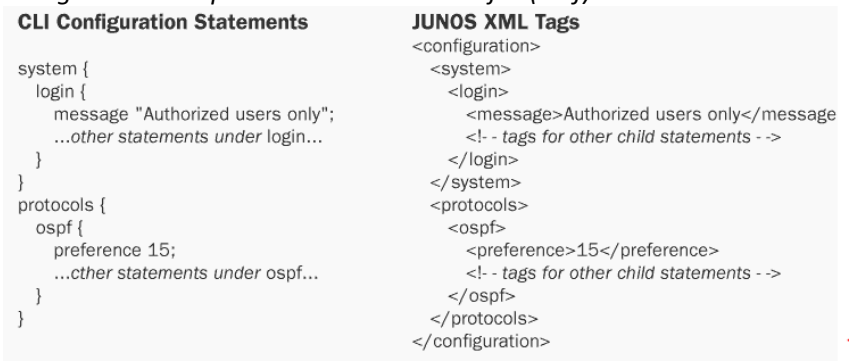
CLI Configuration Statements	JUNOS XML Tags
	<configuration>
protocols {	<protocols>
bgp {	<bgp>
group G1 {	<group>
	<name>G1</name>
type external;	<type>external</type>
peer-as 56;	<peer-as>56</peer-as>
neighbor 10.0.0.1;	<neighbor>
	<name>10.0.0.1</name>
	</neighbor>
}	</group>
group G2 {	<group>
	<name>G2</name>
type external;	<type>external</type>
peer-as 57;	<peer-as>57</peer-as>
neighbor 10.0.10.1;	<neighbor>
	<name>10.0.10.1</name>
	</neighbor>
}	</group>
}	</bgp>
}	</protocols>
	</configuration>

1503

s. Mapeo de declaraciones hijas con un valor único: Las declaraciones hijas denominadas "*leaf*" suelen ser las últimas en la jerarquía y suelen tener un valor asociado en el formato "*keyword value"*. El elemento tag suele tener el mismo nombre del "*keyword"*, y el valor suele encerrarse en dicho

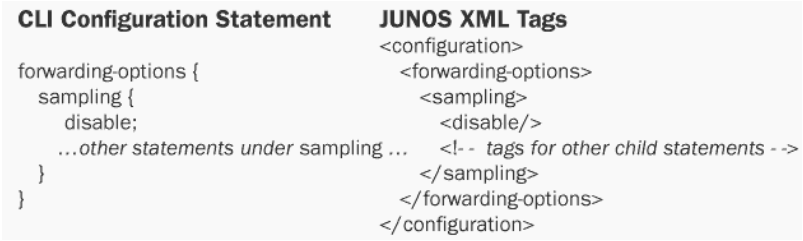
tag. Esto difiere de la sección anterior, “Mapeo de objetos que tienen un identificador” debido a que las declaraciones hijas (leaf) no se repiten en el mismo nivel de jerarquía como ocurre con los objetos (interfaces, grupos de bgp, instancias de ruteo) que poseen un identificador. En la siguiente figura, se ejemplifica lo mencionado.

Figura#57: Mapeo de declaraciones hijas (leaf) con un valor único



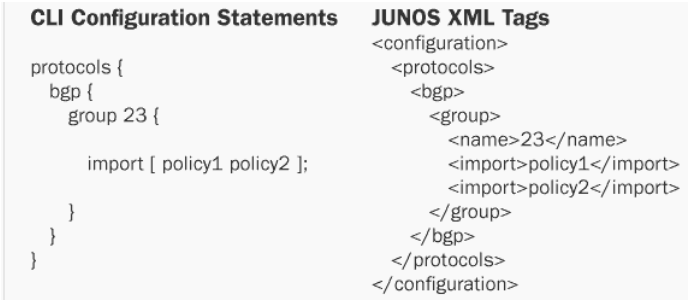
Algunas declaraciones hijas (leaf) poseen únicamente el “*keyword*” sin un valor específico asignado. En esos casos, se asigna únicamente un tag vacío, tal y como se muestra en la figura a continuación.

Figura#58: Mapeo de declaraciones hijas (leaf) sin valor asignado



t. Mapeo de declaraciones hijas con múltiples valores: Las declaraciones con valores múltiples ocurren cuando se asigna una lista de elementos como un conjunto de valores para una declaración hija. El mapeo a Junos XML API ocurre creando un elemento tag por cada elemento de la lista de valores, tal y como se muestra en la figura a continuación.

Figura#59: Mapeo de declaraciones hijas con múltiples valores



u. Uso de RPCs de NETCONF para monitoreo de configuraciones VPLS en Juniper. NETCONF posee RPCs que permiten obtener la información necesaria para el monitoreo de servicios VPLS. Por medio de una aplicación cliente, se realizan las solicitudes deseadas a través de dichos RPCs, se analiza

(parsea) la información dada por el servidor NETCONF del equipo Juniper, de tal forma que se obtengan los datos deseados de forma íntegra. A continuación se resume la jerarquía de RPCs para los comandos "*show vpls mac-table"* y *"show vpls connections"*, básicos en la propuesta de monitoreo para este trabajo.

Figura#60: RPC para el comando "show vpls mac-table"

<get-vpls-mac-table>

Usage

```
<rpc>
  <get-vpls-mac-table>
    <instance>instance</instance>
    <logical-system>logical-system</logical-system>
    <vlan-id>vlan-id</vlan-id>
    <isid>isid</isid>
    <address>address</address>
    <detail/>
    <brief/>
    <extensive/>
    <count/>
  </get-vpls-mac-table>
</rpc>
```

Figura#61: RPC para el comando "show vpls connections"

<get-vpls-connection-information>

Usage

```
<rpc>
  <get-vpls-connection-information>
    <logical-system>logical-system</logical-system>
    <instance>instance</instance>
    <local-site>local-site</local-site>
    <remote-site>remote-site</remote-site>
    <down/>
    <up/>
    <up-down/>
    <brief/>
    <extensive/>
    <history/>
    <instance-history/>
    <status/>
    <summary/>
  </get-vpls-connection-information>
</rpc>
```

4. JUNOS PYEZ. Junos PyEZ es una librería diseñada por Juniper y definida como un micro módulo de Python que permite crear aplicaciones cliente que ejecutan RPCs (disponibles dentro de Junos XML API) y así administrar de forma automatizada y remota equipos que corren el sistema operativo Junos. Está diseñado para proveer las herramientas necesarias para lograr dicho objetivo, permitiendo obtener información de forma equivalente a como se haría a través de una línea de comandos. El nexo común entre Junos PyEZ y Junos OS es NETCONF. Junos PyEZ posee módulos y herramientas internas que le permiten una comunicación por medio del protocolo NETCONF hacia Junos OS. Una gran ventaja de utilizar Junos PyEZ es que provee todas las facilidades del lenguaje de programación Python, un lenguaje ideal para usuarios no familiarizados con el área de programación. En consecuencia, el uso de este módulo le permite al operador obtener la información necesaria para automatización de monitoreo y de configuración sin necesidad de tener

demasiado conocimiento de Junos XML API, Netconf, ni de Junos OS, haciendo uso de las herramientas básicas de Python.

Obtener y ejecutar la información a través de RPCs de Junos XML API, permite realizar un análisis (parseo) más eficiente de la información, al momento de buscar obtener los datos y valores deseados. Junos PyEZ también permite ejecutar comandos de operación y configuración como tal y así obtener la respuesta en el formato ASCII (el normalmente desplegado al usuario), sin la necesidad de la ejecución de RPCs; sin embargo, esto haría el análisis (parseo) de la información obtenida un tanto más complejo y poco escalable. Esto, considerando que futuros cambios en el orden del despliegue de la información en formato ASCII, a través de futuras versiones del sistema operativo Junos, puedan afectar la estructura de la aplicación cliente, obligando su modificación. Como se mencionó anteriormente al describir NETCONF, esto no es inconveniente al analizar (parsear) la información a través de RPCs definidos dentro de Junos XML API.

A continuación, se detalla un breve resumen de las características que ofrece Junos PyEZ.

Características

- Provee información sobre el dispositivo como versión de software, número serial, modelo, etc.
- Responde con datos sobre comandos de operación tanto a través del formato ASCII como del XML.
- Otorga funcionalidades para el análisis de la información obtenida, algo inherente de un lenguaje de programación.
- Permite la configuración automatizada del dispositivo adaptable a condiciones.

Python posee paquetes y módulos de software que permiten crear objetos, ejecutar funciones y realizar análisis de la información dada por el servidor NETCONF del dispositivo Juniper. A continuación se detalla lo más trascendente para este trabajo escrito.

a. Módulos de utilidad en Python

a.1 jnpr.junos: Paquete de software base de la librería Junos EZ, que contiene módulos que permiten la conectividad hacia dispositivos Juniper, así como su administración y configuración remota y automatizada. De entre los de interés destaca el módulo "*device"*.

- device: define la clase "*Device"* y representa al dispositivo que corre el sistema operativo Junos. Contiene métodos para conectarse y conseguir información del mismo, incluyendo para este fin, la vía RPC. Destacan los métodos mencionados a continuación.
    - open: inicia la sesión, a través de NETCONF, con el dispositivo Juniper de interés, manejando parámetros como usuario, contraseña, dirección IP del host, número de puerto, entre otros.
    - cli: ejecuta información de forma idéntica a como se realizaría en una línea de comandos, y la provee respuesta en formato ASCII, es decir, como se le presentaría al usurio que ejecuta el comando. Util cuando no existe el RPC deseado dentro de Junos XML API.
    - close: cierra la sesión a través de NETCONF.
    - facts: devuelve información de fábrica del equipo.

- rpc: ejecuciones equivalentes a los comandos de operación vía RPCs. Se debe recordar que RPC va directamente ligado con el tag de solicitud que se desea realizar.

En la imagen a continuación, se puede observar un ejemplo de programación en Python que muestra el uso del módulo y alguno de sus métodos previamente descritos.

Figura#62: Aplicación cliente utilizando PyEZ

```
from jnpr.junos import Device

dev = Device(host='10.11.12.1', user='user', password='pass', port=22)
get_vpls_macs_rpc = dev.rpc.get_vpls_mac_table()

dev.close()
```

a.2 lxml.etree: Es un módulo de Python que permite trabajar con información obtenida en un formato XML. En pocas palabras, es utilizado para descubrir la información dada por el servidor NETCONF en formato XML y así poder analizarla (parsear). Sus funciones más importantes destacan a continuación.

- fromstring(): Permite convertir una cadena (string) en su equivalente a XML. Se debe tener claro que la cadena debe tener el formato de un elemento tag XML para que funcione.
- tostring(): función útil que convierte una respuesta XML dada por el servidor NETCONF a una cadena guardada bajo una variable.
- Parse(): utilizada para realizar un análisis (parseo) de archivos y objetos XML

En la siguiente figura, puede verse un ejemplo del uso de lxml.etree, utilizando las funciones descritas. El formato XML no es legible para la aplicación cliente, la cual necesita de la función "*tostring()*" usualmente para poder observar la información y realizar el posterior análisis (parseo).

Figura#63: Uso de lxml.etree en Python

```
>>> some_xml_data = "<root>data</root>"

>>> root = etree.fromstring(some_xml_data)
>>> print(root.tag)
root
>>> etree.tostring(root)
b'<root>data</root>'
```

b. Dependencias de Junos PyEZ. A continuación, se describe la función más importante de las herramientas que deben instalarse para poder programar y ejecutar las aplicaciones cliente usando Junos PyEZ. Debe tenerse en cuenta que previamente, una versión de Python igual o mayor a 2.7 (2.7 es la recomendada) debe haber sido instalada.

- Setuptools: es el software "third-party" de Python más importante ya que esto extiende las facilidades para la instalación cualquier otro software necesario en relación a Junos PyEZ. Ez_setup.py es un script que debe correrse para la instalación del setuptools, creando la carpeta scripts, dentro del directorio de Python.

- PIP: es un paquete de software que permite la instalación de otras herramientas de software relacionadas a Junos PyEZ, a través de comandos simples dentro de la línea de comandos de Windows, dentro del directorio scripts.

- PyCrypto: paquete criptográfico de Python que utiliza algoritmos y protocolos, y representa una entidad estructurada que facilita la adición de nuevos módulos.

- Lxml: a través de las dependencias de Setuptools y pip, la instalación del paquete de software lxml se ejecuta a través del comando: pip install lxml, dentro del directorio scripts.

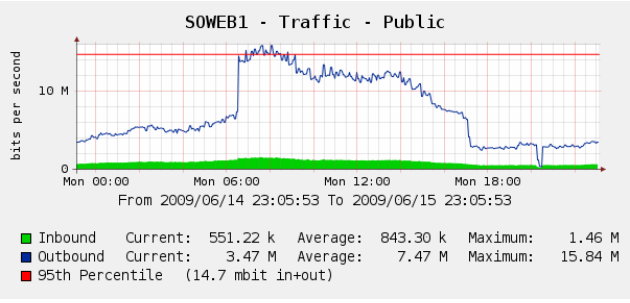
- Paramiko: es una implementación de Python del protocolo SSHv2, proveyendo funcionalidad tanto cliente como servidor. Mientras delibera las tareas de encriptación para paquete de software designado (PyCrypto), Paramiko es en esencia una interfaz de Python orientada al concepto "networking" de SSH. Ejecutado a través del comando: pip install paramiko, desde el directorio correspodiente.

- Junos EZ: es la librería utilizada por Junos PyEZ para la automatización de monitoreo y configuración de dispositivos Juniper. Permite el uso del paquete base de software jnpr.junos para este efecto. Ejecutado a través del comando: pip install junos-eznc.

5. CACTI. Solución open-source completa para la generación de gráficas de red y almacenamiento de datos. Esta herramienta, desarrollada en PHP, provee un servicio de poleo rápido que permite obtener información de equipos remotos y graficar el resultado. Sus plantillas de gráficos son avanzadas, posee múltiples métodos para la recopilación de datos y manejo de usuarios. La interfaz de usuario es simple y fácil de manejar, resultando conveniente tanto para redes a nivel LAN, como redes complejas con cientos de equipos, como lo puede ser una red WAN. Es empleado para obtener gráficas en función del tiempo de métricas como rendimiento de CPU y utilización de ancho de banda. Sin embargo, un uso particular y que es común dentro de empresas de telecomunicaciones es el monitoreo de tráfico por medio de la generación de las gráficas respectivas. Cacti realiza el poleo de equipos de la red vía SNMP (Simple Network Managment Protocol). Por medio de este protocolo, es posible hacer una infinidad de consultas a los dispositivos remotos siempre y cuando estén definidos dentro de una MIB, apuntando a un OID específico y es usual utilizar una fuente de datos, como u script tanto para la recolección de datos como para la generación de gráficas. (SNMP es tratado posteriormente dentro de este marco teórico).

En la siguiente figura se muestra un ejemplo de una gráfica de tráfico haciendo un poleo vía SNMP del dispositivo respectivo:

Figura#64: Gráfica de tráfico en Cacti



A continuación, se detallan aspectos importantes referentes a esta herramienta de gráficos utilizada ampliamente por empresas de telecomunicaciones.

a. Usos comunes. Cacti permite generar casi cualquier tipo de gráfica como una serie temporal, es decir, gráficas en función del tiempo, con la capacidad de generación de gráficas históricas. Dentro de los usos habituales, destacan los que se mencionan a continuación.

- Tráfico de interfaces físicas o virtuales.
- Rendimiento de CPU.
- Temperatura.
- Velocidad de CPU.
- Tiempos RTT de pruebas de ping.
- Jitter de pruebas de ping.

b. Fuente de datos. La recopilación de datos se maneja estableciendo a Cacti, una ruta a scripts o comandos junto con cualquier dato que el usuario necesite ingresar, como lo puede ser nombre del gráfico y de los ejes. Una fuente de datos puede ser también creada, por ejemplo, para medir los tiempos de respuesta de pruebas de ping. En este caso, se crearía un script que haga ping a un host especificando un source específico o indicado por alguna configuración en el equipo (RPM, por ejemplo) y obteniendo otros parámetros necesarios vía SNMP que permitan la generación de la gráfica.

c. Gráficas. Una vez que una fuente de datos es establecida y obtiene los datos, Cacti puede generar la gráfica respectiva. En este sentido, Cacti permite visualizar las gráficas de distintas formas. Entre ellas, destaca vista individual, preliminar, lista de vistas y vista en árbol. Por otro lado, destaca las gráficas de vista históricas, que por medio de acercamientos, pueden mostrar su comportamiento días y horas ocurridos meses atrás.

6. SNMP (SIMPLE NETWORK MANAGMENT PROTOCOL). SNMP es un protocolo de capa de aplicación y utilizado ampliamente para el monitoreo de eventos dentro de equipos de una red IP que requieran asistencia administrativa. Definido dentro del RFC 1157, es un protocolo que ubicado dentro del conjunto definido por el modelo TCP/IP. Los dispositivos que pueden soportar SNMP van más allá de routers y switches, incluyendo servidores, hosts, impresores, entre otros. SNMP expone la administración por medio de variables que identifican un elemento específico dentro del sistema o equipo monitoreado. Estas variables pueden ser requeridas (*query)* por medio de la aplicación cliente o de manejo. El protocolo consiste en tres componentes claves, dispositivos administrados (managed devices), agentes (agents) y sistemas de manejo de red (NMS, Network Managment System), los cuales son tratados a detalle a continuación.

a. Componentes SNMP

- Dispositivo administrado: es un nodo o equipo que posee un agente SNMP y reside dentro de la red administrada. Estos dispositivos pueden ser routers, switches, servidores, computadoras, impresoras, hubs, entre otros, como se mencionó con anterioridad.

- Agente SNMP: módulo de software que reside dentro del dispositivo administrado. El agente traduce la información del dispositivo en un formato compatible con SNMP.

- Sistema de manejo de red (NMS): sistema que corre aplicaciones de monitoreo, en este caso, basadas en SNMP. Proveen los recursos de red y procesamiento requeridos para la red administrada.

b. MIB (Managment Information Base). Es una colección de información organizada jerárquicamente y accedidas por medio de un protocolo como SNMP y que pertenece al dispositivo a ser administrado. Suele ser definida como una jerarquía de información utilizada para definir objetos administrados dentro de un dispositivo de red. Su estructura está basada en forma de árbol, relacionando un grupo de objetos en conjuntos específicos. Cada objeto dentro de una MIB está asociado con un identificador de objeto, OID, el cual le da un nombre único al mismo. Las últimas ramas en la estructura de árbol son las que representan estos objetos, que pueden hacer referencia a un recuso, evento o actividad que ocurra dentro del dispositivo de red administrado. MIB puede ser definida de forma estándar o empresarial. Las MIB estándar son creadas por la IETF y documentadas en varios RFC. Dependiendo del fabricante, muchas MIB estándar son incluídas dentro del software del sistema de manejo de red (NMS). MIB empresariales son desarrolladas y administradas únicamente por equipos específicos del fabricante. En este caso, dichas MIB deben ser obtenidas por medio del fabricante y compiladas en el NMS.

c. OID (Object Identifier). Identificadores de objeto son instancias utilizadas para nombrar unívocamente a los objetos dentro de una jerarquía MIB. Al ser consideradas dentro de la estructura de un árbol, los identificadores de objeto hacen referencia a las ramas finales de la jerarquía, representando a un objeto. Este objeto, como se mencionó puede ser un evento, recurso o actividad que ocurra dentro del dispositivo.

d. Operación SNMP. SNMP trabaja con base en la solicitud de peticiones por parte de los NMS hacia los dispositivos administrados por medio del poleo de los mismos, quienes devuelven una respuesta. Esto se logra con el agente SNMP dentro del equipo como intermediario. La solicitud es implementada utilizando una de cuatro operaciones: *Get, GetNext, Set y Trap.* Los mensajes SNMP consisten en un encabezado y en unidades de datos de protocolo (PDU). El encabezado contiene la versión SNMP y el nombre de la comunidad. Una comunidad es utilizada como una forma de seguridad en SNMP. El PDU depende en el tipo de mensaje que es enviado. Cada dispositivo administrado contiene una base de datos de valores por cada una de las definiciones especificadas jerárquicamente dentro de un MIB. Cada fabricante de dispositivos SNMP posee una sección exclusiva dentro de la estructura de árbol MIB bajo su control. Esta parte es la definida de forma empresarial. De forma que esta estructura sea propiamente organizada, todas las características u objetos administrables de todos los productos de cada fabricante son arreglados dentro del árbol por ramas que se van extendiendo y dividendo en ramas cada vez más específicas. Cada rama creada posee un número y un nombre como identificadores únicos. Cada rama pose un camino completo desde el inicio de la estructura del árbol hasta el punto de interés, formado el nombre de dicho punto. Este identificador es el OID. Como es de imaginar, el inicio de la estructura del árbol es extremadamente general por naturaleza. Conforme se avanza a través de las ramas del mismo se hace más específico.

En la siguiente figura, se muestra el camino completo desde el inicio de la estructura MIB hasta llegar a un OID de interés. Como se observa, esto puede ser especificado tanto por nombre como por valor numérico, siendo de preferencia ésta última opción.

Figura#65: Estructura SNMP

```
Iso(1).org(3).dod(6).internet(1).private(4).transition(868).produc
ts(2).chassis(4).card(1).slotCps(2).-
cpsSlotSummary(1).cpsModuleTable(1).cpsModuleEntry(1).cpsModuleMod
el(3).3562.3

ó

1.3.6.1.4.868.2.4.1.2.1.1.1.3.3562.3
```

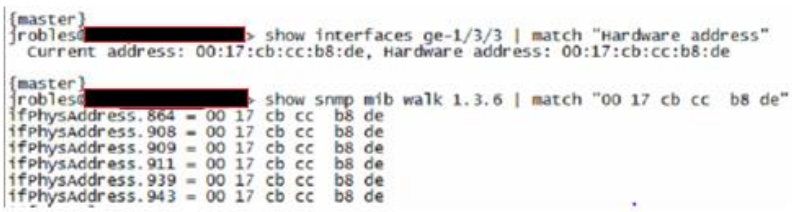
e. SNMP en Juniper. En dispositivos que corren el sistema operative Junos, SNMP ocurre de forma similar a lo explicado con anterioridad. El poleo de dispositivos por medio de este protocolo resulta una opción muy viable para el monitoreo de la red, ya que ofrece una forma rápida y eficaz para ello. La comunicación entre el agente y el NMS ocurre en una de las siguientes formas:

- Solicitudes *GET:* El administrador (NMS) solicita información del agente SNMP, el agente provee una respuesta por medio de una respuesta *GET.*
- Solicitudes *SET:* El administrador (NMS) cambia el valor de un objeto MIB controlado por el agente. El agente indica el estado en una respuesta *SET.*
- Notificaciones *TRAP:* El agente envía trampas para notificar al administrador (NMS) de eventos significativos que ocurren dentro de la red que se monitorea.

Junos OS soporta dos tipos de notificaciones, trampas e informes. Por medio de trampas, el receptor no envía ningún aviso de que ha recibido la trampa. Por lo tanto, el emisor no puede determinar si la trampa fue recibida, ocasionando que la misma pueda perderse debido a un problema ocurrido dentro de la transmisión. Para incrementar la fiabilidad, un informe es similar a una trampa, con la excepción que el informe es almacenado y retransmitido en intervalos regulares de tiempo hasta que ocurre una de dos condiciones. El receptor del informe envía un aviso de recibido al agente SNMP, o un número específico de retransmisiones no exitosas ha ocurrido por lo que el agente descarta el informe. De esta forma es más probable que la transmisión de un informe se complete, haciéndolo más fiable, pero con la desventaja que consume más recursos de la red. Adicionalmente, un informe es almacenado en memoria hasta que ocurra una de las condiciones mencionadas, a diferencia de una trampa SNMP. El uso de informes es recomendado cuando se considera importante que el administrador, NMS, reciba las notificaciones. Sin embargo, si la preocupación habitual es acerca del tráfico de la red, memoria de equipos o envío ICMP, el uso de trampas es la mejor opción.

En la siguiente figura se muestra la dirección física (MAC) de una interfaz de un equipo Juniper. Así mismo, se muestran los correspondientes OID contenidos por el agente SNMP del mismo. El hecho que haya distintos OID, obedece a que la interfaz se compone de distintas interfaces lógicas unívocamente identificadas.

Figura#66: OID SNMP en Juniper



V. METODOLOGÍA

El desarrollo de este Trabajo de Graduación está dividido en cuatro fases distintas, como se muestra a continuación.

A. FASE INICIAL

- SELECCIÓN DEL TEMA: Se escoge el monitoreo de servicios prestados por medio de enlaces MPLS IP-VPN y MPLS-VPLS.

- DELIMITACIÓN DEL TEMA: La delimitación del tema se especifica en el objetivo general y objetivos específicos desarrollados en el protocolo y especificados en este trabajo escrito. El monitoreo se propone realizar sobre 200 enlaces IP-VPN y 50 enlaces VPLS.

- DESARROLLO Y APROBACIÓN DEL PROTOCOLO DE TRABAJO DE GRADUACIÓN: Se detalla lo que se propone realizar durante el periodo del Trabajo de Graduación.

B. FASE DE MONITOREO DE ENLACES MPLS IP-VPN

- INVESTIGACIÓN DE SERVICIOS: Explorar a profundidad la red, documentación de los servicios, casos históricos y apoyo de operaciones locales como último recurso para asegurar la realización del monitoreo de forma correcta.

- MONITOREO RPM: Aplicar el sistema de monitoreo RPM sobre dichos enlaces, definiendo una nomenclatura específica y parámetros estándar.

- GRÁFICAS RPM: Utilizar la herramienta de generación de gráficas para los RPM aplicados.

- SYSLOG: Utilizar el mecanismo para registro de mensajes históricos proveído por el sistema operativo de Juniper, para almacenar incidencias ocurridas con el monitoreo RPM. Esto se busca realizar por medio de archivos syslog y por envío a servidor syslog remoto.

- PLATAFORMA DE ALERTA AL OPERADOR: Además de la plataforma Cacti, se muestra el resultado del monitoreo RPM por medio de una plataforma que clasifica por suscriptor, cada uno de los servicios monitoreados, identificando tipos de alarma por medio de un color específico. Aunque esta plataforma fue realizada por terceros dentro de la empresa, se desea mostrar la utilidad que tuvo la nomenclatura RPM utilizada, así como su utilidad en cumplir con el objetivo de alerta al operador.

- IDENTIFICACIÓN DE INTERFACES: Identificar cada servicio monitoreado de forma estandarizada y ordenada.

- GUÍA ESTANDARIZADA: Documentar la parte del trabajo de monitoreo de capa 3 en una guía corta y de entendimiento rápido para el operador.

C. FASE DE MONITOREO DE ENLACES MPLS VPLS

- INVESTIGACIÓN DE RECURSOS DE MONITOREO: Se busca recopilar la información teórica que permita obtener lecturas de tráfico de las interfaces lógicas asociadas a los PE de las VPLS y reconocimiento de direcciones MAC de las sedes remotas. En otras palabras, se busca determinar las herramientas de monitoreo a utilizar.

- DESARROLLO DEL ALGORITMO DE DIAGNÓSTICO: Se elabora el algoritmo para diagnosticar e identificar y diagnosticar un servicio VPLS de forma automatizada. Para ello, se tiene previamente claro cómo funciona uno de estos servicios y la forma de obtener la información necesaria para garantizar un monitoreo de nivel.

- IMPLEMENTACIÓN DEL ALGORITMO: Se aplica el algoritmo de monitoreo a 50 servicios de esta categoría.

- PRUEBAS Y ANÁLISIS: Se determina la eficacia de la metodología propuesta sobre estos enlaces, al observar cómo responde el monitoreo al momento de tener fallas.

- GRÁFICAS DE TRÁFICO: Utilizando la herramienta de gráficas, se determina visualmente el tráfico que pasa por las interfaces lógicas correspondientes a cada servicio VPLS configurado en los PE.

- DOCUMENTACIÓN: Se realiza la parte del manual estándar para el monitoreo e identificación de estos servicios.

D. FASE FINAL

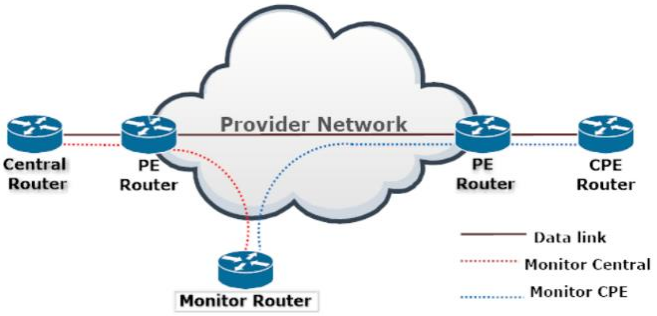
Luego de haber asegurado el correcto funcionamiento del sistema de monitoreo en general, se busca realizar una revisión total de cada aspecto planteado en este protocolo con el fin de corregir y agregar cualquier aspecto que pudiera faltar, así como las correcciones pertinentes al trabajo escrito que se tiene planeado realizar de forma paralela. Esto con la finalidad de que ambas partes sean aprobadas y establecer una fecha de presentación.

VI. DISEÑO EXPERIMENTAL

En esta sección se redactan los pasos a seguir para la implementación del monitoreo en servicios MPLS IP y MPLS VPLS. Cabe destacar que por la naturaleza de los enlaces capa 2 y capa 3, existe una clara diferencia en la aplicación del monitoreo, dividiéndose así en dos grandes apartados. La cantidad de servicios a monitorear por cada capa fue escogida previamente (y mencionada en la metodología). Se llevó un control de los enlaces monitoreados por medio de un archivo de Excel que identifica los aspectos más importantes de los mismos.

A. MONITOREO MPLS IPVPN (CAPA 3)

Figura#67 Diagrama de monitoreo IP



En la Figura 67 se muestra el diagrama del monitoreo de servicios de capa 3, cuyo eje principal fue la aplicación del monitoreo RPM bajo una nomenclatura estricta. Esta nomenclatura obedece a permitir la correcta recolección de los datos para reflejarlos en un portal de monitoreo (realizado por terceros dentro de la empresa de telecomunicaciones). Como un complemento importante, se aplicaron umbrales para generación de mensajes syslog y gráficas de monitoreo en Cacti.

Un análisis profundo sobre la Figura 67, el monitoreo de capa 3 y sus resultados se refleja en la sección de discusión. A continuación se detalla paso por paso la realización de este monitoreo.

1. INVESTIGACIÓN DE SERVICIOS IP. El monitoreo de capa 3 se basa en realizar pruebas de ping a los puntos remotos de los servicios que presta el ISP, incluyendo el punto denominado como la central. El CPE denominado de esta forma representa la sede central del suscriptor y el PE que mira a dicho equipo es llamado PE central. La gestión del PE central está garantizada para fines de este trabajo, no siendo siempre así con los PE remotos. Es necesario el ping a las sedes remotas del suscriptor debido a que los puntos de demarcación generalmente se encuentran ubicados dentro de las instalaciones del mismo, siendo CPEs. Cada servicio destinado a un punto remoto (CPE) está identificado por un número de instalación. En este trabajo, un mismo suscriptor puede tener múltiples sedes remotas (punto-multipunto) o puede únicamente tener dos (punto-punto). En el caso de ser punto-punto se habla únicamente de un número de instalación, ya que la central para fines de este trabajo no se identifica de esa manera. Cuando en el PE que ve hacia la central, se configuran múltiples servicios de un suscriptor en una misma instancia de ruteo VRF, la sonda RPM se compone de varias pruebas de RPM (tests) en función de la cantidad de servicios prestados al suscriptor por dicha VRF, ya que por cada uno, existe un punto remoto o destino.

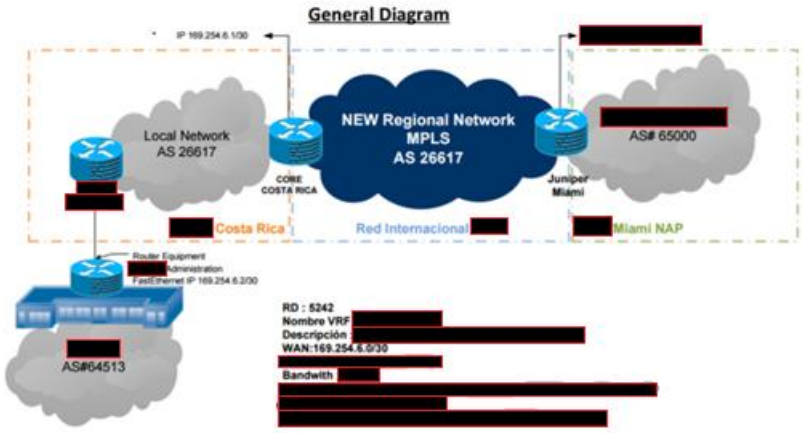
En vista de que para garantizar un monitoreo de nivel es necesario realizar el ping hacia los puntos remotos, se hace también necesario realizar una investigación de cada servicio, por cada número de instalación para obtener las direcciones IP remotas correctas. Esta investigación consiste en revisar la documentación de los servicios, la configuración de los PE central y remotos (a los que se tenga acceso), los casos que se hayan generado por incidencias presentes y pasadas, las copias de configuración de los PE remotos (en caso de tenerlos disponibles) y como última alternativa, consultar con cada operación remota del proveedor por la configuración de la última milla (last mile) de los servicios. Al tener las direcciones IP adecuadas es necesario hacer pruebas de ping y telnet (si es posible) desde el PE central para determinar la conectividad y que las mismas sean las adecuadas. Con esta fase el monitoreo de nivel se garantiza, cubriendo todos los puntos que son responsabilidad del proveedor del servicio (ISP).

A continuación, se describe los pasos que se pueden llegar a realizar durante la fase investigativa del monitoreo de capa 3.

a. Documentación de servicios. La documentación consiste en diagramas del enlace, documentación de bienvenida, pruebas de aceptación del servicio e imágenes de la configuración. Es importante tomar en cuenta que la documentación puede estar incorrecta o haber sufrido algún cambio, por lo que es necesario recurrir a otros pasos que respalden la información.

- Diagramas de enlace: Son útiles ya que dan una persepctiva gráfica de la configuración del servicio. Suele estar detallada con direcciones IP de los puntos finales aunque no siempre es así, y esa información no siempre es la correcta.

Figura#68: Diagrama de enlace IP-VPN



- Documentación de bienvenida: Es la carta entregada al suscriptor y que indica la activación del servicio luego de haber garantizado su funcionalidad. En ella se puede encontrar diagramas del enlace, pruebas de aceptación de servicio, imágenes de configuración y otra información de ayuda.

Figura#69: Introducción de una carta de bienvenida de servicio IP

[redacted] Circuit ID: [redacted]
Your Reference: [redacted]
Circuit Name (reference): [redacted]
Customer: [redacted]
Order Type: Enlace Internacional IP-VPN Elite @ [redacted]
Bandwidth / Speed: [redacted]
End A: [redacted]
End Z: Nap de las Américas MIAMI, FLORIDA, ESTADOS UNIDOS
Ready For Service Date: [redacted]
Billing Start Date: [redacted]

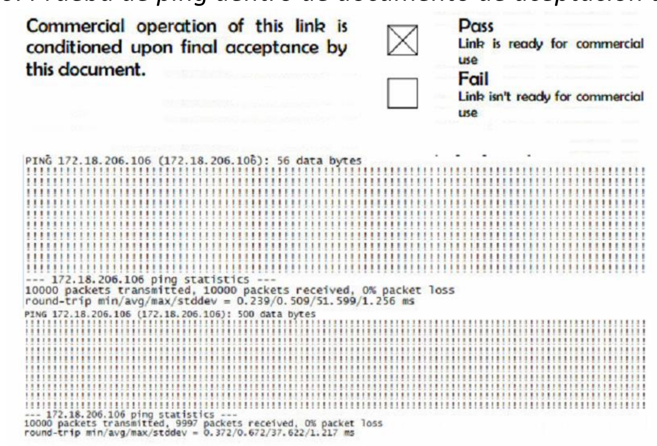
Customer tests and acceptance period starts at mentioned RFS date.

After a 5 day test period, circuit is declared operative and it will be fully managed by our NOC.
You can contact our NOC at: [redacted]

- USA [redacted]
- Central America [redacted]

- Pruebas de aceptación del servicio: Son pruebas que se realizan por un departamento encargado previo a la activación. Entre otras, se puede encontrar pruebas de ping hacia los puntos remotos del servicio. No se garantiza que las direciones IP destino sean las que finalmente se utilizaron.

Figura#70: Prueba de ping dentro de documento de aceptación de servicio



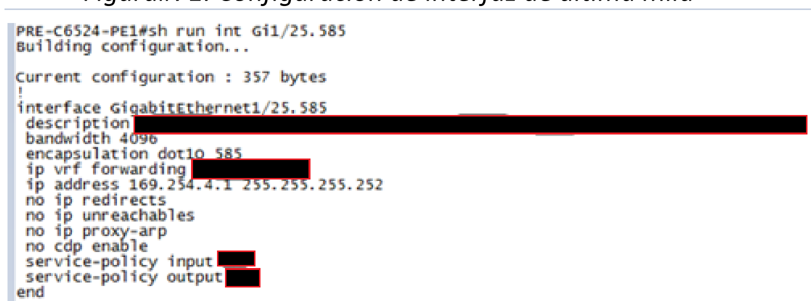
Commercial operation of this link is conditioned upon final acceptance by this document.

☒ Pass — Link is ready for commercial use
☐ Fail — Link isn't ready for commercial use

```
PING 172.18.206.106 (172.18.206.106): 56 data bytes
--- 172.18.206.106 ping statistics ---
10000 packets transmitted, 10000 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.239/0.509/51.599/1.256 ms
PING 172.18.206.106 (172.18.206.106): 500 data bytes
--- 172.18.206.106 ping statistics ---
10000 packets transmitted, 9997 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.372/0.672/37.622/1.217 ms
```

- Imágenes de la configuración: Muestra la configuración del servicio en los PE. De la misma forma, no se garantiza que la información sea la correcta.

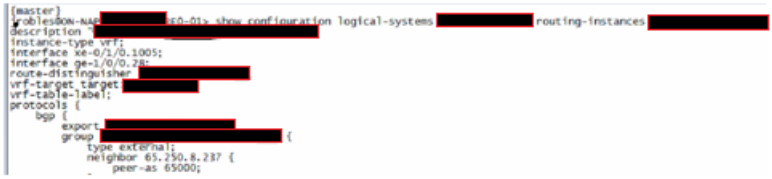
Figura#71: Configuración de interfaz de última mila



```
PRE-C6524-PE1#sh run int Gi1/25.585
Building configuration...

Current configuration : 357 bytes
!
interface GigabitEthernet1/25.585
 description [redacted]
 bandwidth 4096
 encapsulation dot1Q 585
 ip vrf forwarding [redacted]
 ip address 169.254.4.1 255.255.255.252
 no ip redirects
 no ip unreachables
 no ip proxy-arp
 no cdp enable
 service-policy input [redacted]
 service-policy output [redacted]
end
```

b. Configuración PE central y remotos. Revisar la configuración del PE central resulta obligatorio en todos los casos, pues incluye la VRF central, de la cual pueden desprender uno o más servicios de un mismo suscriptor. En ella se indica normalmente una sesión de BGP con el neighbor hacia la sede central del mismo, es decir la dirección IP de la central. Adicionalmente, en la VRF se observan las interfaces involucradas y demás información como el route-distinguisher y vrf-target. Por último, el PE central es al que normalmente se realiza el ruteo para monitoreo RPM desde el equipo de monitoreo destinado para ello.

Figura#72: Configuración VRF de PE central

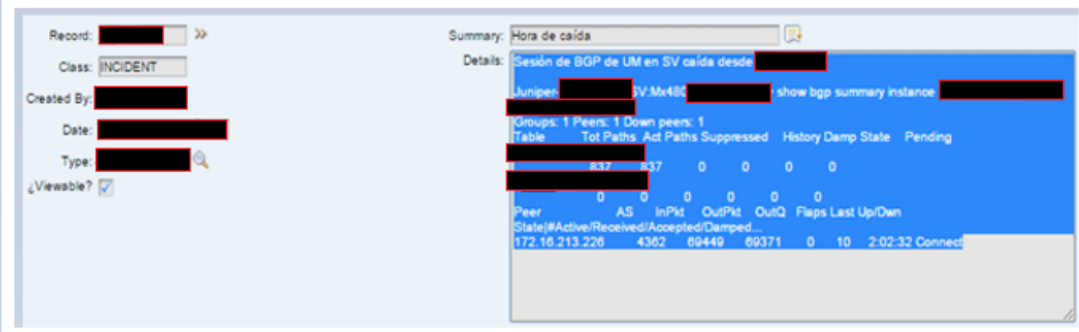


Por otro lado, cuando se tiene gestión de los PE remotos debe realizarse la revisión respectiva. Normalmente se puede encontrar una VRF con sesión BGP directa al CPE de la sede remota, o bien de conexión hacia una red Metro o red MPLS local; es importante discernir esto por medio de descripciones de las interfaces y VRF, números de AS, pruebas de ping y de telnet. Al tener más saltos, debe irse de equipo en equipo si es posible hasta encontrar la red de la última milla.

En la siguiente figura se muestra un ejemplo al revisar un PE remoto (Juniper). Su BGP no está destinada al CPE, sino a un equipo de una red MPLS local (Cisco). Esto se puede verificar por pruebas de telnet, número de AS del vecino BGP, descripción de VRF y subinterfaces. Por telnet se accede al equipo de la red MPLS local (en este caso se tiene acceso), que resulta siendo el equipo de última milla de capa 3, al tener dentro de su VRF, varias sesiones BGP correspondientes a los CPE de diferentes servicios prestados al suscriptor, esto determinado por número de AS. Por descripción de subinterfaces puede distinguirse cada servicio, aunque las mismas pueden no ser complemtamente confiables.

Figura#73 Configuración de PE remoto y equipos de última milla

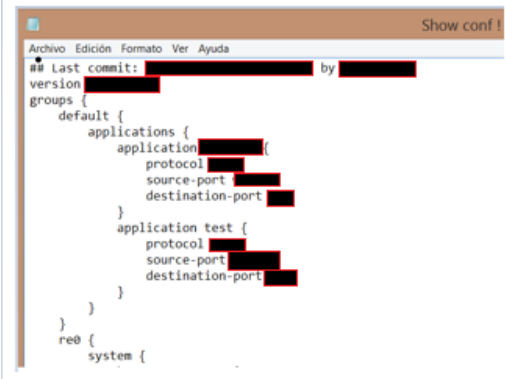
c. Casos generados por incidencias pasadas o presentes. Este paso puede resultar siendo definitivo para encontrar la red de la última milla de los servicios. El operador al llevar a cabo revisiones del servicio afectado, normalmente verifica el estado de la última milla, y coloca en el ticket del caso información sobre la misma. El punto débil de este paso es el error que puede cometer el operador al momento de efectuar revisiones, sin embargo debe revisarse si la descripción de la configuración presentada por dicha persona va acorde al número de instalación del servicio reportado. En la siguiente figura, se puede observar un comentario del ticket sobre un incidente, en el que se brinda el estado de la última milla remota del servicio, y por ende la dirección IP del CPE requerida.



Figura#74: Casos generados por incidencias

d. Copias de configuración de PE remotos: Este método es similar al de la revisión de PE remotos con las claras limitaciones que en realidad no se tiene gestión de dichos equipos. la configuración se presenta en un archivo .txt, .doc, o similar, y las copias pueden estar desactualizadas. En lugar de utilizar la línea de comandos se utiliza el buscador de la aplicación del documento para encontrar información del servicio. Por limitado que sea, esta alternativa puede aportar información correcta.

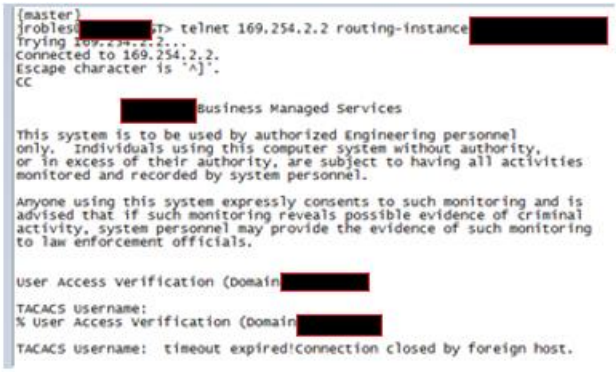
Figura#75: Copias de configuración de PE remotos



e. Consultas a operaciones remotas del ISP. Si ninguno de los pasos anteriores contribuyó a proveer una dirección IP de punto remoto, o si no se tiene la certeza de tener la IP correcta, el último método es solicitar a las operaciones locales, la configuración del servicio IP en equipo de última milla. Esta información debe incluir la VRF (si aplica), las interfaces y sesiones de protocolos asociadas. Puede ocurrir a veces que telnet como forma de prueba no funcione dado que se tienen políticas o filtros por parte del suscriptor que lo impiden.

f. Pruebas de ping y telnet. Al obtener una dirección IP posible a ser la dirección remota deseada, debe realizarse pruebas de ping y telnet. Las pruebas de ping garantizan la conectividad y son elementales para la configuración de RPM. Las pruebas de telnet usalmente permiten confirmar por medio del banner del equipo destino si pertenece o no al CPE. Es probable también que la prueba de telnet no funcione por bloqueos del suscriptor. En ese caso, uno o más de los pasos anteriores debe garantizar que la dirección IP sea la adecuada. En la siguiente figura se muestra cómo haciendo telnet desde un PE remoto se puede garantizar que la dirección IP utilizada es la del CPE en el sitio final. (Por razones de seguridad y privacidad, el nombre del suscriptor se oculta.)

Figura#76: Prueba de telnet hacia un CPE remoto



2. APLICACIÓN DE SERVICIOS RPM. La configuración del servicio RPM se aplicó sobre un equipo de monitoreo que funcionó como dispositivo fuente, hacia los CPE en las sedes remotas involucradas, es decir, los destinos. Para ello, lo primero fue realizar un ruteo entre el equipo de monitoreo y la red del servicio IP. Fue suficiente realizar una conexión hacia el PE que viera hacia la sede central del suscriptor. Luego, se configuró una sonda RPM en el equipo de monitoreo, con las pruebas RPM respectivas, una por cada sede remota del suscriptor. Es importante recalcar que se configuró una sonda RPM por VRF en el PE central. Por cada servicio (distintos números de instalación) que pasara por la VRF se configuró una prueba RPM que apuntara al CPE respectivo. La nomenclatura utilizada en la configuración de sondas también se explica en este apartado.

a. Ruteo para servicio RPM: Los enlaces por defecto no se entregan con un equipo de monitoreo adjunto por parte del ISP para el cual se realizó el trabajo. Esto obliga a crear una conexión entre el equipo de monitoreo y alguno de los equipos que forman parte del enlace a monitorear. Dado que la gestión de equipos PE, del lado de las sedes centrales de los suscriptores, está garantizada y dado que se cuenta con equipos que pueden ser utilizados para monitoreo de ese mismo lado de la red, entonces la conexión se realizó directamente a los equipos PE que apuntan a la sede central del suscriptor.

Cabe destacar que se utilizó un único equipo de monitoreo (router Juniper) que ya tenía conexiones físicas hacia los dos equipos PE por los que pasan los servicios. De ese modo, únicamente se tuvo que trabajar en la conexión lógica, denominada en este trabajo, el ruteo para servicio RPM. Una conexión lógica por cada VRF de los PE centreles. Se debe crear una instancia de ruteo en el equipo de monitoreo y utilizar la VRF respectiva en el PE, unidas entre sí por interfaces lógicas provenientes de las interfaces físicas de conexión. Ambas subinterfaces deben estar dentro de las instancias de ruteo. Por último se debe crear un ruteo estático por default en la instancia del equipo

de monitoreo con *next-hop* igual a la dirección IP de la subinterfaz creada en la instancia del PE para este fin.

El primer paso, previo a realizar estas acciones fue escoger la red de monitoreo. Se decidió utilizar por convención: 172.x.x.0/30, donde la "x" inicia en 1, yéndose en orden sucesivo (172.1.1.0/30, 172.2.2.0/30, 172.3.3.0/30, etc.) por cada VRF de los PE centrales. Se escogió esta convención dado que no es común observarla en las redes de los enlaces. Sin embargo, fue necesario revisar la tabla de rutas por cada servicio al cual se le aplicó el monitoreo RPM para evitar duplicación de IPs. Luego, el procedimiento se resume en los siguientes pasos:

- Creación de las subinterfaces: Ambas subinterfaces fueron creadas para ser instaladas dentro de una instancia de ruteo de capa 3 (VRF, Virtual-Router). Por lo tanto, debieron llevar un vlan-id, una dirección IPv4 y adicionalmente una descripción. El vlan-id fue igual al número "x" más mil, (1000+x), escogido para la identificación de la red.

Se escogió para el equipo de monitoreo, la dirección IP: 172.x.x.2/30. El equipo de monitoreo no usa sistemas lógicos (ver glosario para definición de un sistema lógico) por lo que la configuración se realiza desde fuera.

Figura#77 Creación de subinterfaz en equipo de monitoreo RPM

```
jrobles@jun.m10i[redacted]-re0> show configuration interfaces ge-0/3/3.1001
description "Subinterfaz de equipo de monitoreo RPM";
vlan-id 1001;
family inet {
    address 172.1.1.2/30;
}
```

Se escogió para el equipo PE, la dirección IP: 172.x.x.1/30. Es importante mencionar que dado que todos los servicios en los PE se encuentran dentro de un sistema lógico, la subinterfaz respectiva debe ir dentro del mismo, como se muestra.

Figura#78 Creación de subinterfaz en equipo PE para el RPM

```
{master}
jrobles@OM[redacted]01> show configuration logical-systems [redacted] interfaces ge-1/0/1.1001
description "Subinterfaz de equipo PE para RPM";
vlan-id 1001;
family inet {
    address 172.1.1.1/30;
}
```

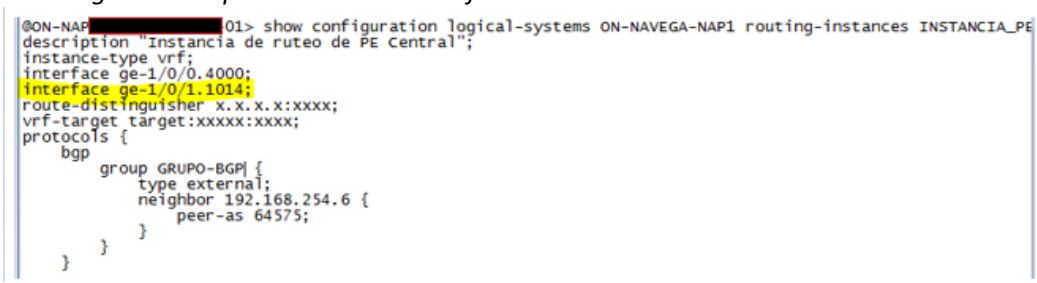
- Creación de instancia de ruteo en equipo RPM: Se crea una instancia de ruteo de conexión para cada VRF ya existente de servicios en los PE. Es decir, se creó una instancia por cada sonda RPM. La misma fue del tipo Virtual-Router y en ella se incluyó la subinterfaz respectiva, el ruteo default y una descripción de la misma.

Figura#79: Creación de instancia de ruteo en equipo RPM

```
@jun.m10i[redacted]02-re0> show configuration routing-instances INSTANCIA-RPM-MONITOREO
instance-type virtual-router;
interface ge-0/3/3.1001;
routing-options {
    static {
        route 0.0.0.0/0 next-hop 172.1.1.1;
    }
}
```

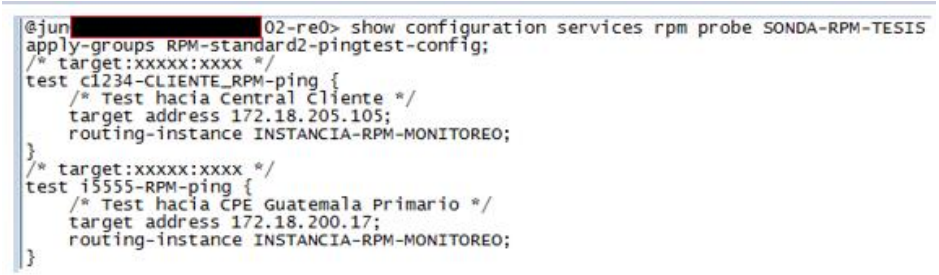
- Aplicación de la subinterfaz de PE en la instancia de ruteo del servicio en PE. Al estar ya creada la instancia de ruteo (VRF) por la cual se transporta uno o varios servicios del suscriptor, únicamente se configuró en ella, la subinterfaz respectiva de conexión al equipo de monitoreo.

Figura#80: Aplicación de subinterfaz en instancia de ruteo de PE



b. Configuración de sondas RPM: La configuración de las sondas RPM constituye una serie de pruebas RPM hacia todas las sedes remotas del servicio prestado. Una vez se realizaron las configuraciones de ruteo previas y se garantizó mediante ping exitosos a los puntos, se procedió a configurar la sonda RPM como tal. La misma tiene parámetros estándar definidos por una configuración predefinida (apply-groups), las pruebas de RPM, las IP destino, la instancia de ruteo fuente y descripciones que permiten identificarlas. Esto, añadido a la nomenclatura utilizada, se podrá ver a continuación (en este ejemplo sólo hay una sede remota y la central que siempre debe existir).

Figura#81: Configuración de una sonda RPM



La configuración de *apply-groups* permite definir parámetros estandarizados para todas las sondas RPM y colocarla sobre la jerarquía *[edit services rpm]*, o jerarquías más específicas. Los únicos parámetros que no son generales son la dirección IP destino y la instancia de ruteo, tal y como puede verse en la figura 6.15. Se definieron en este trabajo tres estándares que varían en la frecuencia con la que realizan las pruebas RPM y de ping. Un estándar u otro se aplica de acuerdo a la criticidad del servicio. Mientras más profundo en la jerarquía se configure uno de estos grupos, más específico es, sobreescribiendo los parámetros repetidos del resto de grupos configurados más arriba en la jerarquía.

- Estándar 1: El nombre y los parámetros definidos en este estándar pueden verse en la figura a continuación. El estándar indica que el RPM es sobre pruebas icmp-ping. Adicionalmente indica que lanza un ping cada 30 segundos y esto lo repite 15 veces. Ocurridas las 15 veces, concluye una prueba RPM (test), espera 150 segundos para iniciar la siguiente. Se configura el peso del ping en 128 bits. Utiliza el umbral de pérdidas sucesivas que por defecto es 1, pero se indica

para enfatizarlo. También utiliza el umbral de pérdida total configurado en 15. Se configuran las trampas SNMP respectivas aunque no se utilizan. La secuencia <*> indica que el parámetro no está definido y depende de las configuraciones individuales. Este estándar es el que se utiliza generalmente y por lo tanto se coloca sobre la jerarquía de *[edit services rpm]*.

Figura#82: Configuración del grupo estándar 1 para RPM

```
{master}
jrobles@[redacted]-re0> show configuration groups RPM-standard-pingtest-config
services {
    rpm {
        probe <*> {
            test <*> {
                probe-type icmp-ping;
                probe-count 15;
                probe-interval 30;
                test-interval 150;
                data-size 128;
                thresholds {
                    successive-loss 1;
                    total-loss 15;
                }
                traps [ test-failure probe-failure ];
            }
        }
    }
}
```

- Estándar 2: En la figura 6.17 puede verse una configuración similar. Dado que esta configuración se aplica a un nivel más específico de jerarquía, únicamente se configuran los parámetros que hacen a este estándar distinto al resto. En este caso solamente fue necesario modificar los dos parámetros de la frecuencia del ping. El ping se realiza cada 10 segundos, durante 15 veces para completar una prueba RPM. Luego, espera 15 segundos y vuelve a empezar. Esta configuración se aplica a sondas RPM específicas o incluso a pruebas de RPM (tests) específicas, a partir de la jerarquía *[edit services rpm probe]*.

Figura#83: Configuración del grupo estándar 2 para RPM

```
jrobles@jun[redacted]02-re0> show configuration groups RPM-standard2-pingtest-config
services {
    rpm {
        probe <*> {
            test <*> {
                probe-interval 10;
                test-interval 15;
            }
        }
    }
}
```

- Estándar 3: Como en el caso anterior, este estándar cambia en la frecuencia del ping del RPM, pero también en la cantidad de ping por prueba RPM. Se realiza un ping por segundo 5 veces para finalizar una de estas pruebas. Luego espera un segundo para repetir. Se tuvo que modificar el umbral de *total-loss* a 5. Su objetivo es atender servicios críticos pudiéndose aplicar a partir de la jerarquía *[edit services rpm probe]*.

Figura#84: Configuración del grupo estándar 3 para RPM

```
{master}
jrobles@[redacted]re0> show configuration groups RPM-standard3-pingtest-config
services {
    rpm {
        probe <*> {
            test <*> {
                probe-count 5;
                probe-interval 1;
                test-interval 1;
                thresholds {
                    total-loss 5;
                }
            }
        }
    }
}
```

c. Nomenclatura de RPM: La nomenclatura utilizada para la identificación de servicios RPM no sirvió únicamente para facilitar el encontrar y revisar las sondas RPM, sino que también aportó para el método de recolección de información de un portal de monitoreo, el cual es creado por terceros dentro de la empresa de telecomunicaciones. La imagen a continuación identifica los puntos clave de la nomenclatura. Luego se explica el fin de cada parte de la misma.

Figura#85: Nomenclatura de RPM

```
probe cCodigoCliente-InstanciaVRF {
    /* target:xxxx:xxxx */
    test iInstalacion1-RPM-ping {
        /* CPE; Guatemala Main */
        target address 10.110.129.102;
        routing-instance INSTANCIAVRF-RPM;
    }
    /* target:xxxx:xxxx */
    test iInstalacion2-RPM-ping {
        /* CPE; Guatemala Backup */
        target address 10.110.129.106;
        routing-instance INSTANCIAVRF-RPM;
    }
    /* target:xxxx:xxxx */
    test cCodigoCliente-Cliente_RPM-ping {
        target address 10.110.130.101;
        routing-instance INSTANCIAVRF-RPM;
    }
}
```

- Nombre de sonda RPM: Se identifica bajo la asignación: *cCodigoCliente-InstanciaVRF.* El código de cliente es el identificador del suscriptor. Permite organizar el monitoreo por cliente en el portal de servicios. Por otro lado, la instancia VRF es el nombre de la instancia de ruteo del PE y su fin es ubicar de forma más sencilla la sonda al momento de diagnosticar. Un ejemplo: c1443-InstanciaCliente, donde el carácter "c" es clave para que el portal de servicios identifique la sonda como tal en la jerarquía *[edit services rpm]*, al momento de la recolección de datos.

- Nombre de prueba de RPM: El nombre de cada prueba o *test* se identifica bajo la asignación *iinstalacion-RPM-ping*, si es una sede remota y le corresponde un número de instalación, o bajo la asignación *cCodigoCliente-Cliente_RPM-ping*, si es la sede central del suscriptor. Los caracteres *"c"* e *"i"* son fundamentales para que el portal pueda identificar si es central o sede remota. El número de instalación permite al operador la identificación rápida del servicio en cuestión. Es importante mencionar que el portal espera que exista una central para tomar en cuenta el monitoreo. Generalmente sólo debe configurarse una.

- Comentarios (Annotates): Un *annotate* es básicamente un comentario que el operador puede agregar en casi cualquier jerarquía. En este caso se utilizan en las jerarquías de pruebas de RPM colocando el target VRF con el siguiente formato: *target:AS-Number:RD*, por ejemplo:

target:23357:3568. Este target es el mismo que se configura en la VRF respectiva en el PE al momento de vender el servicio. Su fin es que el portal pueda identificar y organizar las instalaciones como un conjunto de servicios dados a un suscriptor por una misma VRF.

- Subinterfaces: Adicionalmente, el portal requiere la identificación de subinterfaces en el PE central, de tal forma que pueda determinar las subinterfaces que sean centrales y a qué suscriptor tienen conectividad, dado el código de cliente. También cuantas instalaciones tiene asignada (puntos remotos) cada subinterfaz central. La identificación se da bajo este formato: *cCodigoCliente-DescripciónPaisesDestino (Instalación).* Por ejemplo, una central con dos puntos remotos podría tener esta descripción: "c4344-Cliente GT (12345), HN (23456)".

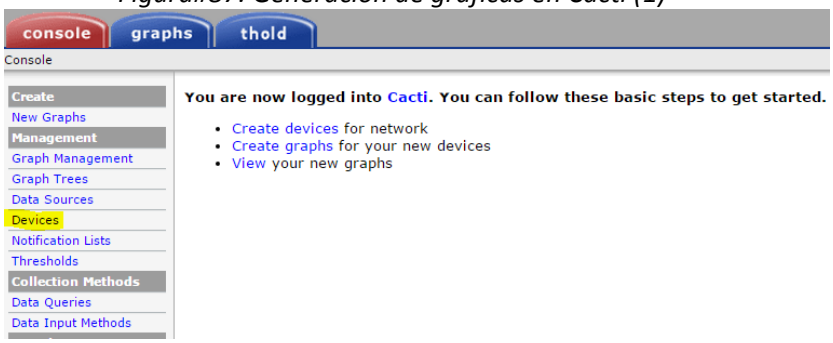
Figura#86: Descripción de subinterfaces

```
@ON-NAP-[redacted]01> show configuration logical-systems [redacted] interfaces ge-1/0/1.61
description "cCodigoCliente-Descripcion pais destino (Instalacion)";
vlan-id 61;
family inet {
    address 68.136.38.150/30;
}
```

3. GENERACIÓN DE GRÁFICAS EN CACTI. La idea de utilizar Cacti es tener registros gráficos históricos del monitoreo RPM. Los scripts necesarios para generar las gráficas ya estaban definidos, por lo que únicamente se utilizaron las herramientas dadas. Como es de esperarse, se generó un gráfico por cada prueba de RPM en cada sonda RPM. A continuación se detallan los pasos para que Cacti refleje gráficas de RPM.

a. Una vez en la página inicial de Cacti, se debe ir a la sección de Dispositivos y seleccionar el dispositivo de interés, en este caso el equipo de monitoreo.

Figura#87: Generación de gráficas en Cacti (1)



b. Ya en el dispositivo, se debe refrescar el Query de RPM para que realice una nueva búsqueda, identificando configuraciones RPM nuevas. Debe verse que el número de Items incremente si es el caso.

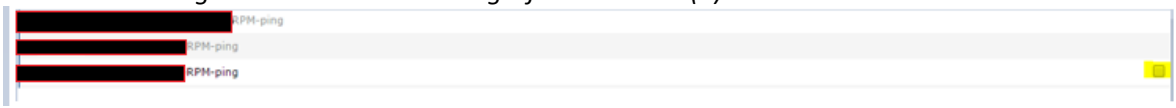
Figura#88: Generación de gráficas en Cacti (2)

c. Debe irse a la sección "*Create Graphs for this Host"* para proceder a la creación del RPM nuevo recién encontrado por el query.

Figura#89 Generación de gráficas en Cacti (3)

d. Se debe buscar el nuevo dato encontrado por el query, marcarlo y dar click en "*create"* en una opción que aparece hasta abajo de la página. Luego de ello, la herramienta comenzará a graficar el servicio.

Figura#90: Generación de gráficas en Cacti (4)



4. SYSLOG. Mediante esta herramienta de registro, el fin es mejorar el diagnóstico realizado por el operador al momento de un incidente. Como se mencionó en el marco teórico, la generación de logs describe unívocamente un evento, permitiendo identificar su ubicación espacial y temporal, el tipo de evento y su descripción. La generación de mensajes syslog se hizo de dos formas distintas. La primera se realizó a través de la generación de archivos syslog en los cuales se almacenan logs considerados de forma temporal, ya que existe un límite en el tamaño y cantidad de registros para el archivo. La segunda fue a través del envío de logs a un servidor syslog. Este método se considera un método permanente de registro pues el espacio para almacenamiento es mucho mayor y especializado para este fin.

a. Syslog por generación de archivos. La configuración se centró en definir ciertos parámetros que se consideraron importantes de acuerdo a los objetivos del uso de esta herramienta. Se configuró el nombre en el formato "rpm_NOMBRE-SONDA-RPM". Se definió utilizar cualquier nivel de recurso y severidad (any any), un tamaño (variable) y cantidad de registros (10) por archivo. El fin fue configurar un archivo por sonda RPM configurada, es decir, por VRF de servicios IP. Esto indica que si para un suscriptor, se tienen 30 servicios que pasan por una VRF del PE central, el archivo syslog se utiliza para almacenar eventos de esos 30. El "match" fue necesario para delimitar e indicar la sonda RPM y el tipo de evento al identificarlo por el código de mensaje, cuyos eventos se almacenarán en el mencionado archivo. Se pidió generar un log por cada ping perdido de una prueba RPM, y un log por cada prueba de RPM fallida en un cien por ciento (Los umbrales respectivos se definieron en la configuración de las sondas RPM, como se vio con anterioridad en este marco). Por último, se definió el parámetro "world-readable" para que pueda ser visto por cualquier operador.

A continuación puede verse el formato de configuración utilizado para la generación de archivos syslog.

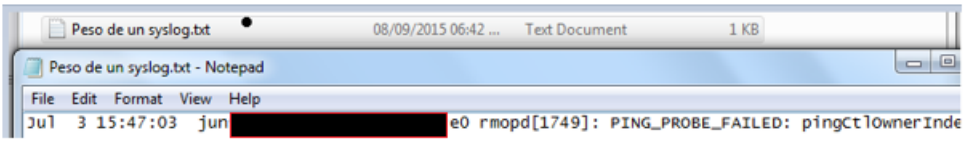
Figura#91: Formato de configuración syslog

```
@jun[redacted]e0> show configuration system syslog
file rpm_NOMBRE-SONDA-RPM {
    any any;
    match "PING_PROBE_FAILED: pingCtlOwnerIndex = NOMBRE-SONDA-RPM";
    match "PING_TEST_FAILED: pingCtlOwnerIndex = NOMBRE-SONDA-RPM";
    archive size 1m files 10 world-readable;
}
```

Se definió como estándar 10 registros por archivo y un tamaño en función de la cantidad de servicios por VRF, es decir, la cantidad de pruebas RPM por sonda RPM. Se definió un tamaño de 1MB por servicio o instalación monitoreada por el archivo, basado en el peso de un log y la capacidad disponible del equipo de monitoreo. Es decir que para la sonda RPM con 30 pruebas RPM (es decir, 30 instalaciones o servicios monitoreados), el tamaño configurado fue de 30MB, pero al tener capacidad de 10 registros, la capacidad máxima sería de 300MB.

Al ser el tamaño de un log de 1KB, se tiene la capacidad de almacenar 1000 logs por registro de archivo por servicio. Tomando en cuenta que son 10 registros por archivo, se define una capacidad de 10,000 logs por cada servicio monitoreado por un archivo syslog, es decir 10MB.

Figura#92: Peso de un mensaje syslog



Tomando en cuenta el monitoreo de 200 servicios diferentes, se estaría utilizando una capacidad de 2GB (200*10MB) del directorio */var/*. Se debe recordar que syslog almacena mensajes dentro del directorio */var/log/.*

Figura#93: Capacidad de almacenamiento de directorio /var/

```
jrobles@jun[redacted]e0> show system storage  | match var
Filesystem             Size       Used      Avail  Capacity   Mounted on
/dev/ad1s1f             34G       9.1G        22G       29%    /var
```

Puede observarse de la figura 6.27 que el uso de 2G representa únicamente el 5% de la capacidad, teniendo inicialmente disponible el 71%. Adicionalmente, se comprobó la ubicación de archivos syslog dentro del directorio */var/log/*.

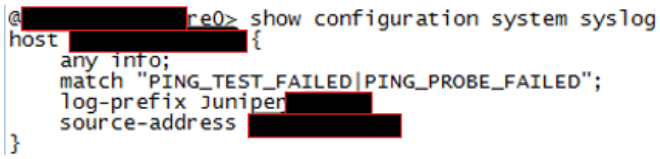
Figura#94: Almacenamiennto de archivos syslog en directorio /var/log/

```
jrobles@jun.m10i.nap.mia.dat.02-re0> file list /var/log/ | match rpm_
RPM_CLIENTE1
RPM_CLIENTE1.0.gz
RPM_CLIENTE1.1.gz
rpm_CLIENTE2
rpm_NOMBRE-SONDA-RPM1
rpm_NOMBRE-SONDA-RPM2
rpm_NOMBRE-SONDA-RPM3
```

b. Syslog por envío a servidor. El envío de mensajes syslog al servidor especializado para este efecto, fue más sencillo, dado que únicamente se realizó una configuración especificando la IP de un host específico, una prioridad dada por “any info”, un match que delimita el envío de logs generados por los umbrales configurados para las sondas RPM

(“PING_TEST_FAILED|PING_PROBE_FAILED"), un log-prefix usado para identificar el equipo de envío, y la IP fuente (en este caso la IP de gestión del equipo de monitoreo).

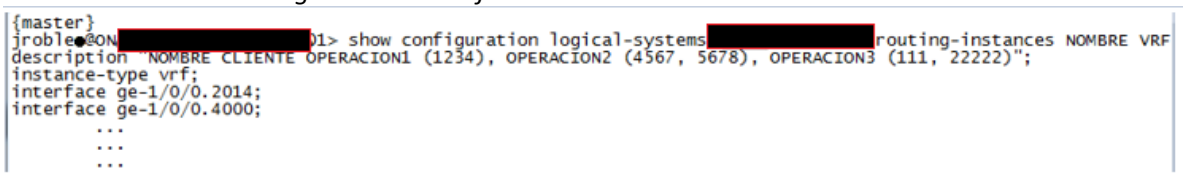
Figura#95: Syslog por envío a servidor



5. IDENTIFICACIÓN DE VRFs y SUBINTERFACES. El fin de la identificación de VRFs y subinterfaces fue facilitar el diagnóstico del operador al momento de la revisión de incidentes. Esto se realizó en las descripciones de las VRFs y subinterfaces del PE central. La clave para realizar dicha identificación fue incluir la instalación de todos los servicios que pasaran tanto por la VRF central como la interfaz lógica central, y los países u operaciones por instalaciones, en las descripciones respectivas. La identificación de interfaces centrales también tuvo el fin de clasificarlas por cliente o suscriptor para el portal de servicios.

a. VRFs. Se aplicó la identificación en la descripción de la VRF respectiva. Se siguió el estándar: “Nombre Cliente OPERACIÓN (Instalación1, instalación2…), donde se especificó el nombre del suscriptor, la operación o país destino de los servicios de dicho cliente, y los números de instalación de los servicios por país. Así como se observa en la figura, los servicios de un suscriptor que pasan por la VRF del PE central pueden tener destino a operaciones diversas, y pueden tener varios servicios o instalaciones por país.

Figura#96: Identificación de servicios en VRFs



b. Subinterfaces. La identificación fue aplicada en la descripción de la subinterfaz central para cada VRF monitoreada en el PE central. Se consideró que dentro de una VRF pueden haber diferentes interfaces asociadas por diversas razones (conexión a central de suscriptor, conexión de equipos Core del ISP, de monitoreo de ISP, de conexión directa a destino final, etc.). Por lo tanto, es importante identificar primero la subinterfaz que tiene conexión hacia la central. Una forma es observando la descripción que se le colocó (si se le colocó) a la misma al momento de la activación de los servicios, pero la forma más segura es verificando que la IP de la subinterfaz sea la contraparte del neighbor o vecino de BGP configurado en la VRF. Esto aprovechando que las conexiones hacia las sedes centrales de los suscriptores ocurre típicamente mediante el establecimiento de una sesión BGP.

Figura#97: Determinación de la subinterfaz central en VRFs

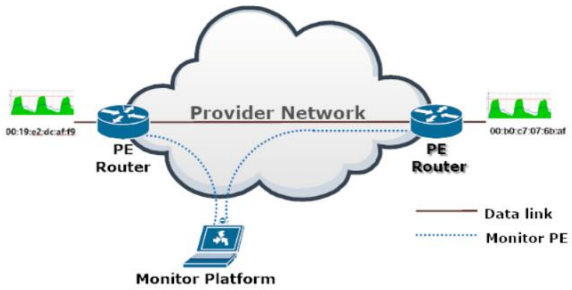
```
{master}
jrobles@ON-[redacted]-01> show configuration logical-systems [redacted] routing-instances NOMBRE VRF
description "NOMBRE CLIENTE OPERACION1 (1234), OPERACION2 (4567, 5678), OPERACION3 (111, 22222)";
instance-type vrf;
interface ge-1/0/0.2014;
interface ge-1/0/0.4000;
interface ge-1/0/1.1014;
route-distinguisher 10.20.134.1:8193;
vrf-target target:26617:8193;
vrf-table-label;
protocols {
    bgp {
        group GRUPO-BGP {
            type external;
            neighbor 10.20.30.1 {
                peer-as 64575;
            }
        }
    }
}

{master}
jrobles@ON-[redacted]-01> show configuration logical-systems ON-NAVEGA-NAP1 interfaces ge-1/0/0.4000
description "DESCRIPCION DEL SERVICIO";
vlan-id 4000;
family inet {
    address 10.20.30.2/30;
}
```

Luego de haber determinado qué subinterfaz es la central, se aplicó la identificación de la misma bajo el formato: "*cCodigoCliente-Nombre Cliente OPERACIÓN (instalación1, instalación2…)".* El hecho de colocar el formato de identificación de código de cliente con una "c" antepuesta fue para la recolección y clasificación de la información de centrales por cliente o suscriptor, dentro del portal de servicios. En el caso de presentarse más de una central (esto por respaldo, típicamente), simplemente se identificaron aclarando la diferencia entre una u otra.

Figura#98: Identificación de subinterfaces centrales

```
{master}
jrobles@ON[redacted]-01> show configuration logical-systems [redacted] interfaces ge-1/0/0.4000
description "c321-NOMBRE CLIENTE OPERACION1 (1234), OPERACION2 (4567, 5678), OPERACION3 (111, 22222)";
vlan-id 4000;
family inet {
    address 10.20.30.2/30;
}
```

B. MONITOREO MPLS VPLS (CAPA 2)



Figura#99: Diagrama de monitoreo Ethernet

En la Figura 99 puede verse el diagrama del monitoreo implementado que refleja los dos componentes básicos del mismo, la lectura de tráfico y el reconocimiento de direcciones MAC (bajo el cual se implementó un algoritmo de diagnóstico). En este caso interesó obtener la información de tráfico en las interfaces de salida de la red MPLS de los equipos PE, y la tabla de direcciones MAC de los servicios VPLS del lado del PE central. No resulta trascendente la información que los equipos CE puedan proporcionar. Con el hecho que en lugar de un router como equipo de monitoreo, se

coloque una laptop, se trata de indicar que la información fue obtenida por herramientas ajenas (PyEZ, Cacti) a un dispositivo de capa 3, y que son de uso general de un host u ordenador.

A continuación, se desarrollan los pasos realizados que llevaron al planteo, desarrollo e implementación de este sistema de monitoreo.

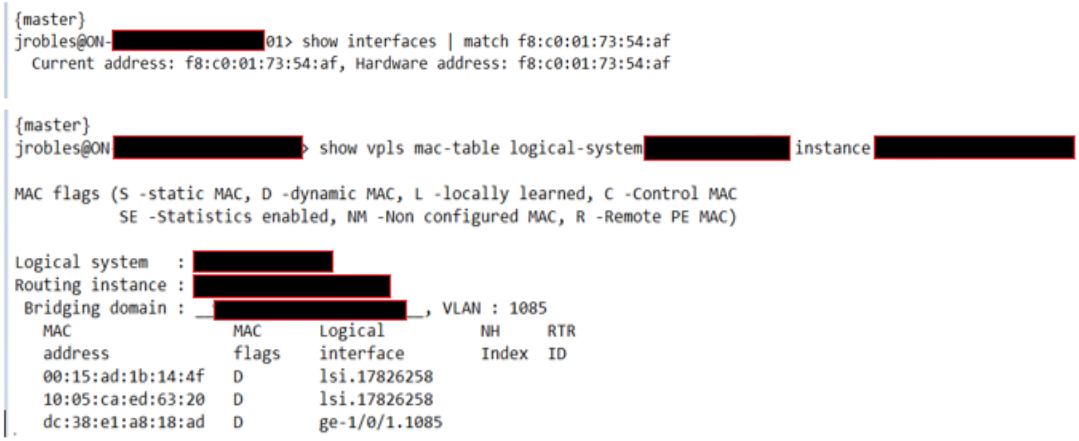
1. INVESTIGACIÓN DE RECURSOS DE MONITOREO: Por la naturaleza de servicios Ethernet, el monitoreo no puede realizarse vía ping por iniciativa sólo del proveedor. Es necesario solicitar al suscriptor brindar una direccionamiento IP (típicamente CIDR /29) para colocar una IP en el equipo de monitoreo. Esta opción se consideró secundaria y aplicable sólo a enlaces críticos, ya que no es escalable por depender del suscriptor. Por lo tanto, se determinó realizar el monitoreo por gráficas de tráfico y el desarrollo de un algoritmo que confirme la caída de los servicios VPLS. Este algoritmo, basado en el reconocimiento de las direcciones MAC de la tabla de los servicios.

Una vez se tuvo clara la idea del monitoreo de los servicios VPLS, se buscó determinar con qué herramientas resultaría factible realizarlo. El protocolo SNMP permite obtener información de los servicios por medio de sus OIDs y sin ningún tipo de autenticación. El protocolo NETCONF permite obtener y configurar cualquier información que esté incluida dentro del JUNOS XML API, solicitada por RPCs. Sin embargo, NETCONF requiere del establecimiento de una sesión y posterior autenticación, lo cual hace más lento el proceso. Aun así, La principal desventaja del uso de SNMP es que no existen OIDs para identificar las MAC de la tabla VPLS de los servicios.

La etapa de investigación se conformó de tres partes: determinar que vía SNMP no pueden leerse direcciones MAC, utilizar NETCONF y PyEZ para obtener esta información y determinar la herramienta para graficar tráfico, definiendo los puntos de lectura.

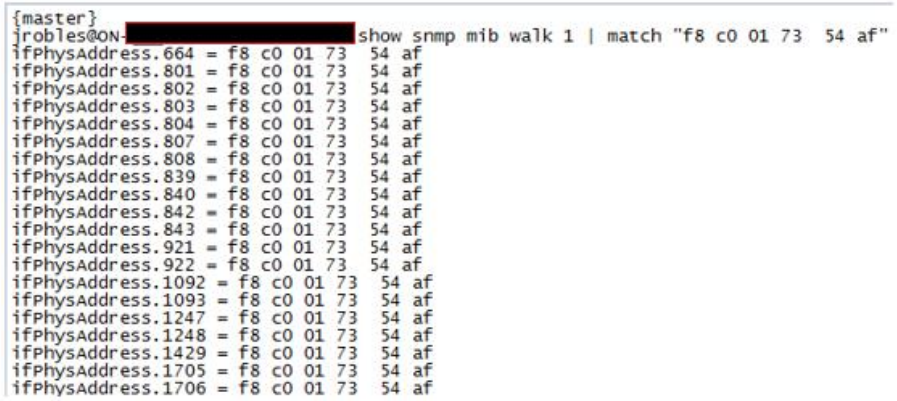
a. Recurso SNMP. Se identificó la dirección MAC de una interfaz física y las direcciones MAC de una VPLS en un equipo PE Juniper. (Los servicios VPLS están instalados en los dos PE que son Juniper).

Figura#100: Direcciones MAC de una interfaz física y de una VPLS



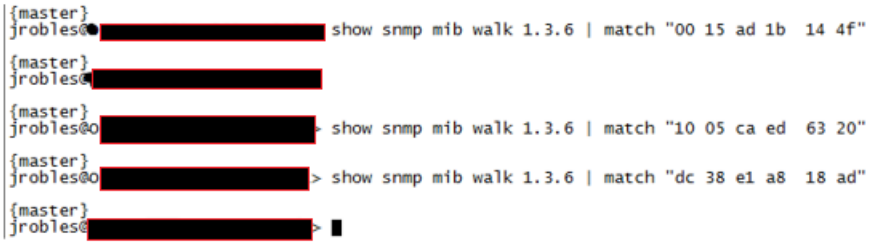
Se comprobó que vía SNMP, la MAC de la interfaz física posee su propio OID. En la imagen puede observarse que para cada interfaz lógica de la mencionada interfaz física, existe un OID distinto.

Figura#101: OIDs de las subinterfaces de una interfaz física



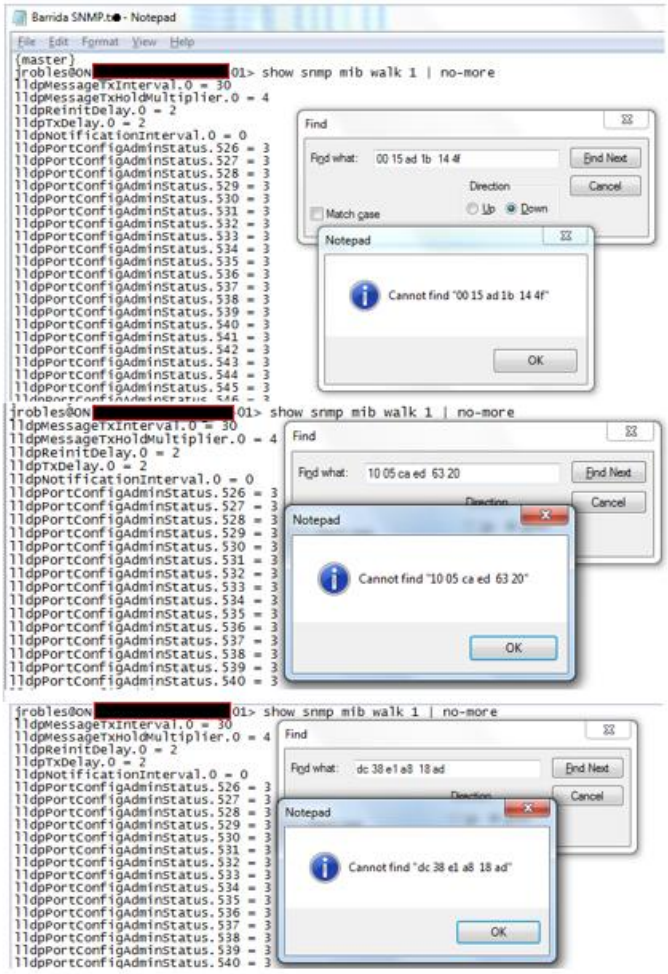
Se comprobó que vía SNMP, las MACs de la VPLS no poseen OID. El equipo realiza la búsqueda (la cual tarda varios minutos) y luego no despliega ningún resultado.

Figura#102: Búsqueda de OIDs para MACs de un servicio VPLS



Finalmente se barrió todo el equipo en búsqueda de todos los OIDs del mismo. Se guardó en un archivo .txt, y como puede verse en la imagen, no se encontró ninguna de las tres direcciones MAC del servicio.

Figura#103: Búsqueda de MACs de VPLS en un barrido SNMP



b. Recursos NETCONF y PyEZ. Al ser NETCONF un protocolo, no sólo para obtener información de equipos Juniper sino también para configurar el mismo (siempre y cuando existan los RPCs respectivos dentro del Junos XML API), el mismo se presenta como una herramienta factible para obtener la información requerida, analizar (parsear) los datos deseados e implementar el algoritmo para diagnosticar los servicios VPLS. Esto de la mano con PyEZ y sus dependencias, que son las herramientas de programación para este fin.

Se determinaron las librerías y módulos necesarios para la importación de variables y funciones necesarias. De la librería JNPR.JUNOS, se importó el objeto DEVICE, para la creación de una variable que se autenticara con el equipo y los métodos que ofrece para obtener información del mismo. De la librería LXML se importó el objeto ETREE, utilizado para convertir la información dada por el servidor NETCONF en formato XML a formato STRING. Por último, la librería SYS fue utilizada para realizar una excepción y terminar correctamente el programa al darse problemas en la etapa de establecimiento de sesión y autenticación.

Figura#104: Liberías de Python utilizadas para la implementación del algoritmo

```
from jnpr.junos import Device
from lxml import etree
import sys
```

NETCONF requiere inicialmente establecer una sesión segura y autenticarse con el equipo mediante un usuario configurado en el mismo o vía un método de autenticación (RADIUS, TACACS). Para ello es necesario crear una variable del objeto DEVICE y especificar el hostname del equipo. En este caso, se utilizó la dirección IP de gestión del mismo. Así mismo, se especificó el usuario, la clave respectiva y el puerto de capa de aplicación. A pesar de que se debe habilitar NETCONF y configurar el puerto 830 en el equipo, el reconocimiento únicamente funciona a través del puerto 22 (SSH). Se utilizó el método *open()* del objeto DEVICE y se encerró en un *try/except* para asegurar la correcta finalización del programa. Esto se logró incluyendo en la excepción, el objeto SYS con el método EXIT y parámetro 1, usado para especificar que la salida es porque ocurrió un error en el programa. El parámetro 0 es usado para indicar una salida limpia, pero no es de interés en este caso.

Figura#105: Autenticación al servidor NETCONF del equipo vía SSH.

```
#Definicion DEVICE
dev = Device(host='[redacted]', user='jrobles', password='[redacted]', port=22)
#Login a DEVICE
try:
 dev.open()
except Exception as err:
 print err
 sys.exit(1)
```

A continuación se muestra cómo verificar y realizar la habilitación de NETCONF en el dispositivo.

Figura#106: Habilitación de NETCONF en dispositivo Juniper

```
{master}
jrobles@O[redacted]> show configuration system services netconf | display set
set system services netconf ssh port 830

{master}
jrobles@ON[redacted]01> show configuration system services netconf
ssh {
    port 830;
}
```

Para poder diagnosticar adecuadamente un servicio VPLS, dos llamadas de procedimiento remoto (RPCs) son necesarias. Una que permita obtener la tabla MAC del servicio, y la otra, el estado de la conexión del mismo. Utilizando la variable del objeto DEVICE se obtuvo esta información y se almacenó en variables respectivas. Los métodos son *"get-vpls-mac-table()"* y *"get-vpls-connection-information()"*. Esto se determinó verificando en el equipo mediante los comandos especificados en el marco teórico.

En la siguiente figura, se muestra cómo determinar esta información.

Figura#107: Determinación de los RPCs necesarios para el diagnóstico VPLS

```
{master}
jrobles@ON-[redacted]01> show vpls connections | display xml rpc
<rpc-reply xmlns:junos="http://xml.juniper.net/junos/12.3R7/junos">
    <rpc>
        <get-vpls-connection-information>
        </get-vpls-connection-information>
    </rpc>
    <cli>
        <banner>{master}</banner>
    </cli>
</rpc-reply>

{master}
jrobles@ON[redacted]01> show vpls mac-table | display xml rpc
<rpc-reply xmlns:junos="http://xml.juniper.net/junos/12.3R7/junos">
    <rpc>
        <get-vpls-mac-table>
        </get-vpls-mac-table>
    </rpc>
    <cli>
        <banner>{master}</banner>
    </cli>
</rpc-reply>
```

A continuación, se muestra cómo utilizar Python haciendo uso de los módulos necesarios para solicitar esa información al servidor NETCONF del equipo. En este caso, como los servicios VPLS se almacenan dentro de un sistema lógico especificado para ello, dentro de los parámetros de ambos métodos debe hacerse esta aclaración. Se utiliza el método TOSTRING() del objeto ETREE para convertir la información en formato XML a formato STRING y así poder realizar el análisis (parseo) necesario.

Figura#108: RPC´s enviados al servidor NETCONF utilizando Python

```
get_vpls_macs_rpc = dev.rpc.get_vpls_mac_table(logical_system ='[redacted]')
get_vpls_macs_tostring = etree.tostring(get_vpls_macs_rpc) #El resultado del GET
```

Para verificar cómo instalar las herramientas de monitoreo utilizadas para el monitoreo de servicios de capa 2 sobre un ordenador con sistema operativo Windows, favor referirse a la sección de anexos.

c. Recurso Cacti. En el monitoreo de capa 3, Cacti fue utilizado como herramienta para generar las gráficas RPM, en las que se requirió el RTT y el Jitter. Sin embargo, como se mencionó en el marco teórico, Cacti es comúnmente utilizado para la generación de gráficas de tráfico. En el monitoreo de servicios VPLS, el tráfico es una herramienta fundamental como primer paso para determinar si un servicio es candidato a estar fuera de servicio y ser evaluado por el programa que implementa el algoritmo. Se definió que se debía graficar el tráfico de las interfaces lógicas de salida de la red MPLS en los PE involucrados, tanto del PE central, como de los PE remotos por cada servicio. Esto debido a que si un PE central tiene varios PE remotos asociados, graficar solo del central no garantiza que todos los servicios estén operativos. Como se ha mencionado con anterioridad, se tiene gestión garantizada del PE central. Sin embargo, no ocurre así con los puntos remotos. Por ello, fue necesario solicitar a las operaciones su colaboración con permitir la lectura SNMP de sus dispositivos y así poder graficar el tráfico de las interfaces lógicas asociadas a cada servicio. Se especificó la creación de gráficos históricos (24 horas, semanal, mensual y anual), al igual que para las gráficas de RPM.

2. DESARROLLO DEL ALGORITMO DE DIAGNÓSTICO. Previo al desarrollo del programa utilizado para diagnosticar un servicio VPLS, fue necesario determinar el algoritmo en el que está basado. El algoritmo es la serie de pasos a implementar para cumplir con el objetivo y ofrecer un diagnóstico de nivel como parte del monitoreo de servicios VPLS. Para ello, es necesario estar familiarizado con cómo funciona un servicio VPLS y cómo normalmente un operador lo diagnostica.

a. Funcionamiento de un servicio VPLS. Como se explicó en el marco teórico, un enlace VPLS emula una conexión Ethernet montada sobre una red MPLS. En la analogía realizada, se define la red MPLS como el switch al cual se conectara los usuarios, los PE son los puertos de dicho switch y los equipos CE son los hosts que se conectan a través dichos puertos. En este caso, la cantidad de sitios es la cantidad de puertos disponibles del equipo y es necesario que se reciban MACs de todos los puntos disponibles para determinar la operatividad del servicio. Cuando un dispositivo dentro del servicio levanta una conexión IP, su MAC aparece en la tabla de la VPLS. En ese sentido, este tipo de conexiones las realiza el suscriptor. El proveedor puede realizarlo con fines de monitoreo. Es necesario evaluar las conexiones del servicio con el fin de determinar la cantidad de sedes locales y remotas. Se debe tener especial cuidado con identificar qué interfaces lógicas locales representan sedes distintas y cuáles son de respaldo de una misma sede. En el caso de interfaces lógicas remotas ("LSI"), se comprobó que únicamente se recibe una por PE remoto sin hacer diferenciación de las varias interfaces lógicas que pueden estar configuradas en dicho PE remoto. En el caso de las locales, sí se puede hacer esta diferenciación. Esto limita el diagnóstico si un sitio VPLS remoto posee varias sedes asociadas. Es normal que cuando una conexión VPLS tiene estado "Dn" (no operativa), no se reciban direcciones MAC remotas, por lo que requerir del equipo esta información resulta innecesario si la base del diagnóstico es la revisión de la disponibilidad de estas direcciones. Si se reciben MACs de todos los sitios del servicio, es probable que el servicio esté operativo, pero puede que hayan MACs intermedias reconocidas por tema de monitoreo. Es por ello que el algoritmo desarrollado tiene pasos para diagnosticar el servicio a mayor profundidad. El tema de recibir sólo una LSI por PE remoto (sin diferenciar las interfaces que pueden representar distintas sedes) representó una limitante de entrada y no fue tratado en el algoritmo.

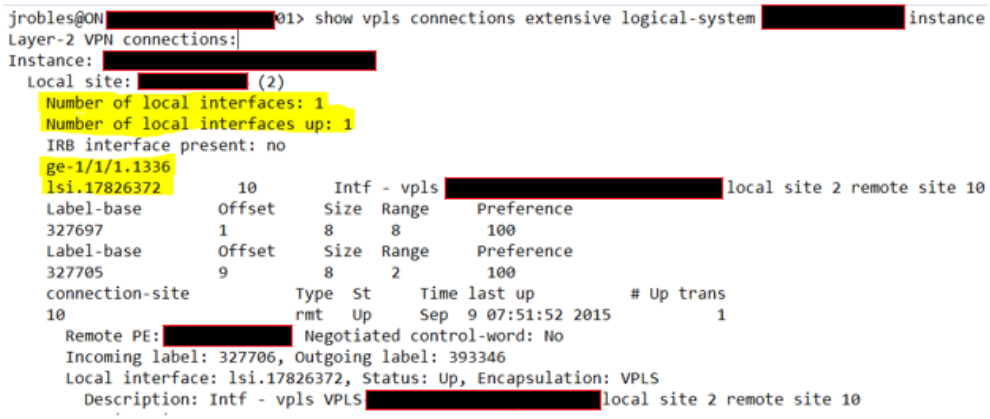
b. Diagnóstico del operador. El diagnóstico normal del operador consiste en dos pasos esenciales. El uso del comando "show vpls connections" con el fin de verificar el estado de la VPLS y el uso del comando "show vpls mac-table". Sin embargo, se muestran pasos intermedios, algunos de los cuales, son necesarios para el desarrollo del algoritmo utilizado. A continuación se muestra el uso del primer comando en mención.

Figura#109: Despliegue del comando "show vpls connections"

```
jrobles@ON[redacted] show vpls connections logical-system[redacted] instance VPLS-[redacted]

Instance: [redacted]
  Local site: [redacted] (2)
    connection-site           Type  St     Time last up          # Up trans
    10                        rmt   Up     Sep  9 07:51:52 2015           1
      Remote PE: [redacted], Negotiated control-word: No
      Incoming label: 327706, Outgoing label: 393346
      Local interface: lsi.17826372, Status: Up, Encapsulation: VPLS
        Description: Intf - vpls VPLS-[redacted]local site 2 remote site 10
```

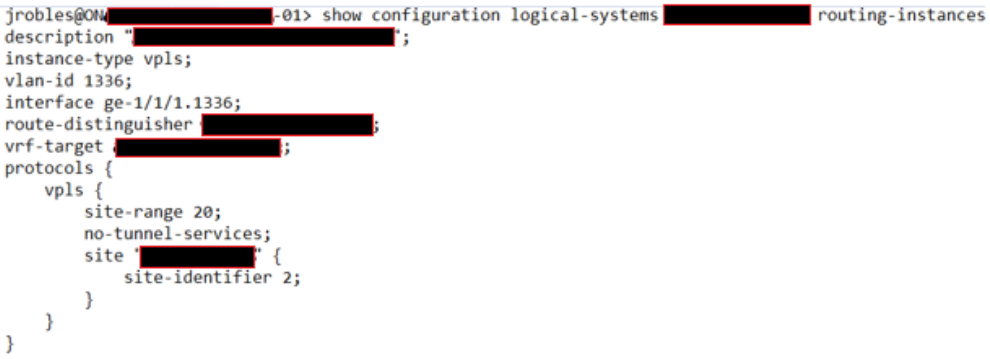
Si la conexión VPLS se encuentra Up se procede a la revisión de tabla de direcciones MAC. Sin embargo el comando *"show vpls connections extensive"* es un paso intermedio que no se aplica comúnmente. El mismo permite verificar la cantidad de interfaces lógicas locales y remotas. Puede verse en el caso de la figura a continuación, que se trata de un servicio VPLS punto a punto.

Figura#110: Despliegue del comando "show vpls connections extensive"



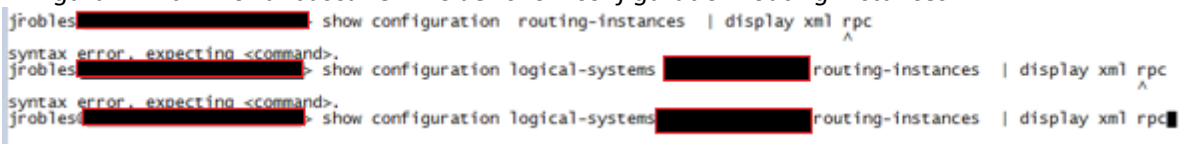
Para verificar si las interfaces locales representan distintas sedes o el respaldo de una misma sede, se debe utilizar el comando *"show configuration routing-instances Nombre-VPLS"*.

Figura#111: Despliegue del comando "show configuration routing-instances VPLS"



Se puede ver que no existe un RPC para el comando de configuración, por lo que fue necesario obtenerlo del método CLI de la clase DEVICE.

Figura#111b: Error al buscar el RPC de "show configuration routing-instances"



A continuación puede verse cómo se comprobó la diferenciación entre interfaces lógicas locales activas y de respaldo, y así determinar cantidad de sedes (servicios) locales. Se observa en un PE la configuración y conexión de un servicio VPLS. En el despliegue de las conexiones se observan dos interfaces lógicas locales y una remota, sin embargo no se hace distinción de cuáles están activas. Con el despliegue de la configuración, se indica que una interfaz lógica local es primaria y la otra de respaldo por medio del parámetro *"active-interface primary"*. De esta forma se comprobó que sí es posible hacer diferenciación entre las interfaces lógicas locales activas y de respaldo. No sucede lo mismo en el caso de las remotas, como se verá más adelante.

Figura#112: Revisión de servicio VPLS (Interfaces locales)

```
{master:member0-re0}
jrobles@[redacted] show configuration routing-instances [redacted]
description [redacted];
instance-type vpls;
vlan-id 1263;
interface xe-1/0/1.1263;
interface xe-13/0/1.1263;
no-local-switching;
route-distinguisher [redacted]
vrf-target target:[redacted]
protocols {
    vpls {
        site-range 20;
        no-tunnel-services;
        site site100 {
            site-identifier 3;
            active-interface primary xe-1/0/1.1263;
            interface xe-1/0/1.1263;
            interface xe-13/0/1.1263;
        }
    }
}

{master:member0-re0}
jrobles@[redacted] show vpls connections extensive instance [redacted]
Layer-2 VPN connections:

Instance: [redacted]
  Local site: site100 (3)
    Number of local interfaces: 2
    Number of local interfaces up: 2
    IRB interface present: no
    xe-1/0/1.1263
    xe-13/0/1.1263      |
    lsi.1052537          1         Intf - vpls [redacted] local site 3 remote site 1
    Label-base        Offset     Size  Range      Preference
    262449            1          8     8          100
    connection-site           Type  St     Time last up          # Up trans
    1                         rmt   Up     Aug 14 12:07:29 2015           1
      Remote PE: [redacted], Negotiated control-word: No
```

Se comprobó que cuando un sitio local no tiene interfaz configurada como activa primaria, se pueden recibir direcciones MAC de todas las interfaces locales asociadas. Se asume así, distintas sedes. En cambio, cuando se tiene configurada una interfaz como activa primaria, sólo de ella se recibe direcciones MAC, ignorando las otras. Se asume en este caso, una sede asociada al sitio.

Figura#112.b: Efecto de interfaz local activa primaria (1)

```
@[redacted]01> show configuration logical-systems [redacted] routing-instances VPLS-PRUEBA
instance-type vpls;
vlan-id 1318;
interface ge-1/0/0.324;
interface ge-1/0/1.324;
interface ge-1/1/1.324;
route-distinguisher [redacted];
vrf-target target:[redacted];
protocols {
    vpls {
        site-range 20;
        no-tunnel-services;
        site [redacted] {
            site-identifier 1;
            active-interface primary ge-1/1/1.324;
        }
    }
}

@[redacted]-01> show vpls mac-table logical-system [redacted] instance VPLS-PRUEBA
Logical system    : [redacted]
Routing instance : VPLS-PRUEBA
 Bridging domain : __VPLS-PRUEBA__, VLAN : 1318
   MAC                 MAC      Logical          NH     RTR
   address             flags    interface        Index  ID
   00:23:9c:94:f7:f0   D        lsi.68165386
   74:a0:2f:97:8e:40   D        lsi.68165386
   74:a0:2f:c5:e2:e7   D        ge-1/1/1.324
```

Figura#112.c: Efecto de interfaz local activa primaria (2)

```
@C[redacted]01> show configuration logical-systems [redacted] routing-instances VPLS-PRUEBA
instance-type vpls;
vlan-id 1318;
interface ge-1/0/0.324;
interface ge-1/0/1.324;
interface ge-1/1/1.324;
route-distinguisher [redacted];
vrf-target target:[redacted];
protocols {
    vpls {
        site-range 20;
        no-tunnel-services;
        site [redacted] {
            site-identifier 1;
        }
    }
}

@C[redacted]01> show vpls mac-table logical-system [redacted] instance VPLS-PRUEBA
Logical system   : [redacted]
Routing instance : VPLS-PRUEBA
 Bridging domain : __VPLS-PRUEBA__, VLAN : 1318
   MAC                 MAC      Logical          NH     RTR
   address             flags    interface        Index  ID
   00:17:cb:cc:b8:db   D        ge-1/0/0.324
   00:1f:12:d4:38:60   D        ge-1/0/1.324
   00:23:9c:94:f7:f0   D        lsi.68165386
   74:a0:2f:97:8e:40   D        lsi.68165386
   74:a0:2f:c5:e2:e7   D        ge-1/1/1.324
```

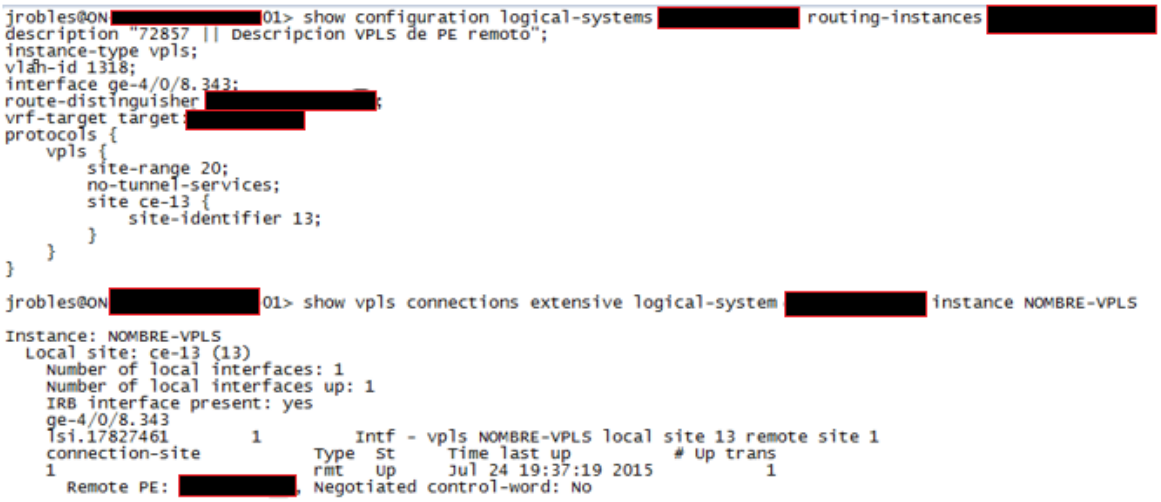
Del lado de un PE central, se observa, para un servicio VPLS, un sitio asociado únicamente. El PE central posee tres interfaces lógicas locales y las tres activas, ya que en la configuración no se especifica ningún tipo de redundancia. Esto indica tres sedes locales (servicios o instalaciones) asociadas. Se observa una interfaz remota.

Figura#113.a: Revisión de un servicio VPLS (Interfaces remotas) (1)

```
jrobles@ON[redacted]-01> show configuration logical-systems [redacted] routing-instances NOMBRE-VPLS
description "Nombre VPLS (72857)";
instance-type vpls;
vlan-id 1318;
interface ge-1/0/0.324;
interface ge-1/0/1.324;
interface ge-1/1/1.324;
route-distinguisher [redacted]
vrf-target target[redacted]
protocols {
    vpls {
        site-range 20;
        no-tunnel-services;
        site MX-960[redacted] {
            site-identifier 1;
        }
    }
}

jrobles@ON[redacted]01> show vpls connections extensive logical-system [redacted] instance [redacted]

Instance: NOMBRE-VPLS
  Local site: MX-960[redacted] (1)
    Number of local interfaces: 3
    Number of local interfaces up: 3
    IRB interface present: no
    ge-1/0/0.324
    ge-1/0/1.324
    ge-1/1/1.324
    lsi.68165386        13        Intf - vpls NOMBRE-VPLS local site 1 remote site 13
    connection-site           Type  St     Time last up          # Up trans
    13                        rmt   Up     Jul 24 19:37:19 2015           1
      Remote PE: [redacted] Negotiated control-word: No
      Local interface: lsi.68165386, Status: Up, Encapsulation: VPLS
        Description: Intf - vpls NOMBRE-VPLS local site 1 remote site 13
```

Se observa en el PE remoto que la interfaz LSI vista desde el PE central, corresponde únicamente a una interfaz local en dicho PE remoto. Sin embargo, se puede observar en este PE que, a pesar que el central tiene tres interfaces activas, las identifica bajo el mismo LSI. Esto llevó a la conclusión que existe una LSI por sitio, sin importar el número de sus sedes asociadas. Al no haber forma de hacer distinción, esto represento una limitante a tomar en cuenta en la elaboración del diagnóstico. (Se puede comprobar que sí se revisó en los dos PE, dado que en el central el identificador del sitio es 1, mientras que en el remoto es 13.)

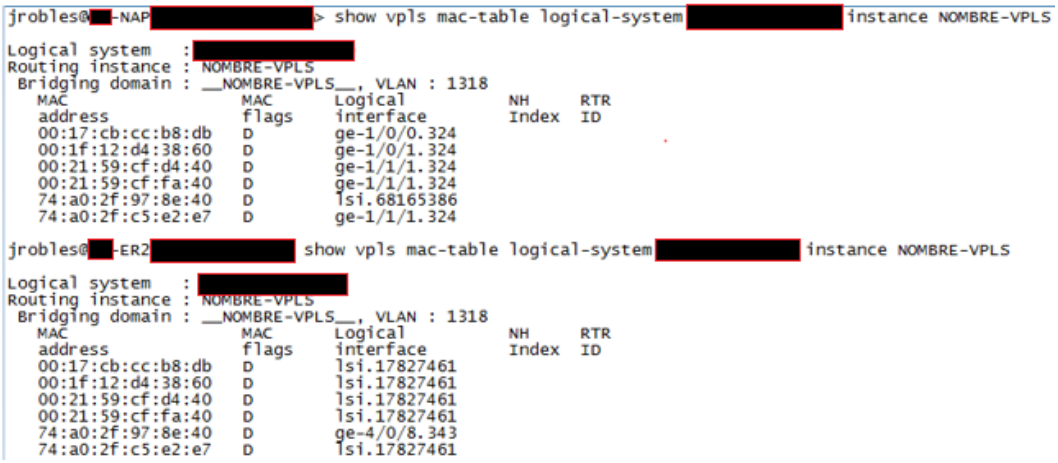
Figura#113.b: Revisión de un servicio VPLS (Interfaces remotas) (2)



```
jrobles@ON[redacted]01> show configuration logical-systems [redacted] routing-instances [redacted]
description "72857 || Descripcion VPLS de PE remoto";
instance-type vpls;
vlan-id 1318;
interface ge-4/0/8.343;
route-distinguisher [redacted];
vrf-target target:[redacted]
protocols {
    vpls {
        site-range 20;
        no-tunnel-services;
        site ce-13 {
            site-identifier 13;
        }
    }
}

jrobles@ON[redacted]01> show vpls connections extensive logical-system [redacted] instance NOMBRE-VPLS

Instance: NOMBRE-VPLS
  Local site: ce-13 (13)
    Number of local interfaces: 1
    Number of local interfaces up: 1
    IRB interface present: yes
    ge-4/0/8.343
    lsi.17827461        1         Intf - vpls NOMBRE-VPLS local site 13 remote site 1
    connection-site           Type  St     Time last up          # Up trans
    1                         rmt   Up     Jul 24 19:37:19 2015           1
      Remote PE: [redacted], Negotiated control-word: No
```

En las tablas MAC de la VPLS para ambos sitios (PE), se confirma que se identifica la misma LSI por sitio remoto sin importar sus interfaces activas y por ende dejando la cantidad de sedes (servicios) asociados a sitios remotos como indeterminada. (Esto sabiendo que el servicio VPLS estaba totalmente operativo en ese momento.)

Figura#113.c: Revisión de un servicio VPLS (Tablas MAC)



```
jrobles@[redacted]-NAP[redacted]> show vpls mac-table logical-system [redacted] instance NOMBRE-VPLS

Logical system   : [redacted]
Routing instance : NOMBRE-VPLS
 Bridging domain : __NOMBRE-VPLS__, VLAN : 1318
   MAC                 MAC      Logical          NH     RTR
   address             flags    interface        Index  ID
   00:17:cb:cc:b8:db   D        ge-1/0/0.324
   00:1f:12:d4:38:60   D        ge-1/0/1.324
   00:21:59:cf:d4:40   D        ge-1/1/1.324
   00:21:59:cf:fa:40   D        ge-1/1/1.324
   74:a0:2f:97:8e:40   D        lsi.68165386
   74:a0:2f:c5:e2:e7   D        ge-1/1/1.324

jrobles@[redacted]-ER2[redacted] show vpls mac-table logical-system [redacted] instance NOMBRE-VPLS

Logical system   : [redacted]
Routing instance : NOMBRE-VPLS
 Bridging domain : __NOMBRE-VPLS__, VLAN : 1318
   MAC                 MAC      Logical          NH     RTR
   address             flags    interface        Index  ID
   00:17:cb:cc:b8:db   D        lsi.17827461
   00:1f:12:d4:38:60   D        lsi.17827461
   00:21:59:cf:d4:40   D        lsi.17827461
   00:21:59:cf:fa:40   D        lsi.17827461
   74:a0:2f:97:8e:40   D        ge-4/0/8.343
   74:a0:2f:c5:e2:e7   D        lsi.17827461
```

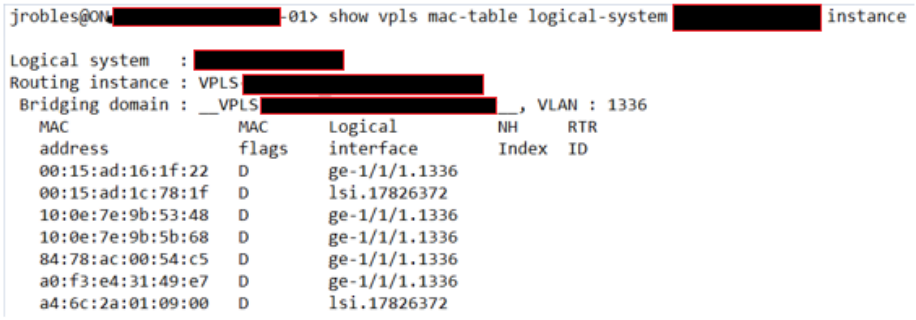
Además aunque no se muestre toda la salida del comando en la Figura 113.b, en la Figura 113.a puede verse claramente que en la descripción del sitio remoto se le asocia la LSI 68165386 que es de la que el PE central recibe direcciones MAC como se ve en la Figura 113.c (primera tabla MAC). En la Figura 113.b se cortó la salida que asocia el sitio remoto de ese PE a la LSI 17827461. Esto refuerza el hecho que existe una misma LSI por sitio remoto.

Si la conexión está caída, puede verificarse la conexión BGP, la aplicación de políticas y comunidades relacionadas al servicio. Como se mencionó, para el fin del algoritmo basado en la disponibilidad de MACs, la revisión del estado de la conexión se hace innecesaria. Sin embargo, se describen tres pasos básicos para dicha revisión:

- Determinar y revisar la configuración BGP hacia el/los PE remotos. Normalmente puede desarrollarse dentro de un grupo BGP por PE remoto, en el sistema lógico. Con ello, identificar las políticas de importación y exportación.
- Revisar la configuración de las políticas y verificar que la comunidad VPLS (usualmente del mismo nombre que la instancia de ruteo) esté incluida en las mismas.
- Revisar que la configuración de la comunidad coincida con el *vrf-target* tanto del PE local como del o los remotos.
- Revisar que se anuncie y reciba la comunidad asociada por cada PE remoto, haciendo match por el *vrf-target.*

El segundo paso del operador es la ejecución del comando *"show vpls mac-table"*. En el mismo el operador revisa que existan MACs provenientes de todos los puntos. Se reitera que el hecho que esto se cumpla, no garantiza la operatividad completa del servicio VPLS, ya que no se tiene la garantía que todas las direcciones MAC sean del suscriptor; algunas por tema de monitoreo IP, pueden ser del proveedor. Para ello el algoritmo debe almacenar un registro de MACs por servicio barrido al detectarlo operativo por primera vez. Luego debe comparar la cantidad y determinar cambios de MAC en un barrido actual con las almacenadas en el registro. Con base en eso, debe determinar un resultado de diagnóstico específico. (El caso de no discernir cantidad de sedes por sitio remoto, es una limitante que no se pudo cubrir.)

Figura#114: Despliegue del comando "show vpls mac-table"



c. Algoritmo a implementar. Acorde a lo mencionado en las secciones recientes, el algoritmo se compone de los siguientes pasos, dividido en dos partes. La de la detección y análisis (parseo), y la de diagnóstico.

i. Detección y "parseo" de VPLS Mac Tables:

i.i. Se ejecuta el RPC para obtener todas las tablas MAC de las VPLS dentro del sistema lógico.
i.ii. El resultado del RPC se convierte en String y se guarda en una variable independiente.
i.iii. Se determina la cantidad de VPLS Mac Tables por medio del tag element "l2ald-mac-entry".
i.iv. Por medio del mismo tag element se hace una lista que separa las tablas MAC por servicio VPLS en sus índices. El tamaño de la lista, en función de la cantidad de tablas detectadas.

i.v Ciclo general para la evaluación individual de tablas MAC detectadas:
i.v.i Se establece una condición para evaluar solo en los índices impares de la lista. Por cada ciclo, se evalúa un índice, que corresponde a la tabla MAC de una VPLS.
i.v.ii Se determina la cantidad de MACs por servicio con el tag element: l2-mac-address.
i.v.iii Se realiza el parseo de identificación del servicio:
i.v.iii.i Se obtiene el nombre de la VPLS con el tag element: l2-mac-routing-instance.
i.v.iii.ii Se obtiene el nombre del sistema lógico con el tag element: l2-mac-logical-system.

i.v.iii.iii Se obtiene el vlan-id con el tag element:

i.v.iv Detección y "parseo" de las conexiones de la VPLS en análisis:

i.v.iv.i Se ejecuta el RPC para obtener las conexiones de la VPLS en evaluación.

i.v.iv.ii El resultado del RPC se convierte en String y se guarda en una variable.

i.v.iv.iii Se utiliza el tag element "label-block-heading" para eliminar texto no deseado en el posterior parseo o análisis de las conexiones.

i.v.iv.iv Se utiliza el tag element "interface-name" para determinar la cantidad de interfaces de las conexiones de la VPLS.

i.v.iv.v Se hace una lista con el mismo tag element para iniciar el parseo de cada interfaz, asociada en sus índices. El tamaño de la lista en función de la cantidad de interfaces de las conexiones.

i.v.iv.vi Ciclo en función de la cantidad de interfaces de las conexiones de la VPLS:

i.v.iv.vi.i Condición para evaluar sólo en los elementos impares de la lista, en donde se ubica la información de cada interfaz asociada.

i.v.iv.vi.ii Se completa el parseo de las interfaces y se guarda en lista.

i.v.iv.vii Ciclo en función de la cantidad de interfaces de las conexiones de la VPLS:

i.v.iv.vii.i Se determina la cantidad de interfaces remotas.

i.v.iv.vii.ii Se determina la cantidad de interfaces locales.

i.v.iv.viii Si existen dos o más interfaces locales asociadas al servicio VPLS, Se determina por el comando de configuración de la VPLS, si existe una interfaz activa primaria configurada, para sólo tomar esa en cuenta (ignorar el resto).

i.v.iv.ix Se realiza una lista donde se reúne la información de cantidades totales, remotas y locales de las interfaces de las conexiones de la VPLS.

i.v.v Se realiza el parseo para las MAC e interfaces detectadas de la tabla VPLS en turno.

i.v.v.i Parseo inicial de direcciones MAC con el tag element: l2-mac-address.

i.v.v.ii Parseo inicial de subinterfaces con: l2-mac-logical-interface.

i.v.v.iii Ciclo en función de la cantidad de direcciones MAC del servicio:

i.v.v.iii.i Condición para evaluar solo en los elementos impares.

i.v.v.iii.iiSe completa el parseo de direcciones MAC e interfaces asociadas.

i.v.vi Se despliega la información global del servicio VPLS.

II. Diagnóstico:

ii.i Se procede a revisar el servicio actual del barrido realizado por el RPC de MACs según el ciclo general.

ii.ii Si se detectan menos MAC Addresses que interfaces lógicas asociadas a las conexiones del servicio VPLS, el mismo se detecta no operativo.

ii.ii.i Variable booleana "service_down" es True.

ii.ii.ii Variable booleana "mac_count_less" es True.

ii.iii Se revisa que direcciones MAC provengan de todas las interfaces asociadas de conexión por medio de un doble ciclo, utilizando las listas creadas.

ii.iii.i Mientras el contador 1 sea menor a la cantidad total de interfaces de conexión:

ii.iii.i.i Mientras el contador 2 sea menor al total de interfaces detectadas en la tabla MAC:

ii.iii.i.i Evalua si la interfaz de conexión en turno es igual a la interfaz de la tabla MAC en turno

ii.iii.i.i.i Si es igual, variable de presencia de interfaz en True y se sale del subciclo.

ii.iii.i.i.ii Si es distinta, variable de presencia de interfaz en False e incrementa en uno el contador 2 para comparar la misma interfaz de conexión con la próxima interfaz detectada de tabla MAC, si existe. Si no, se sale del subciclo.

ii.iii.i.ii Si la variable de presencia de interfaz es True:

ii.iii.i.ii.i Actualiza el índice correspondiente de la lista booleana de control de pérdida de interfaces como False.

ii.iii.i.ii.ii Si no, deja el índice en True.

ii.iii.i.iii Se reinicia contador 2 y aumenta en uno el contador 1 para evaluar la siguiente interfaz de conexión, si existe. Si no, se sale del subciclo.

ii.iv Se realiza un ciclo para barrer la lista booleana de control de pérdida de interfaces:
ii.iv.i Si se detecta al menos un índice como "True", entonces service_down = True.
ii.iv.ii Si todos los índices se detectan en "False", entonces service_down = False.

ii.v Realiza asignación de variables permanentes (registro) para el control de mac_addresses y mac_count únicamente si detecta servicio operativo y si esto no ha sido realizado con anterioridad, o si se determina necesario actualizar el registro por medio de la variable booleana "get_vpls_update". Si no se realiza la asignación y ésta no se ha realizado con anterioridad, las evaluaciones del paso 11 y 12 no cumplirán su objetivo, ya que compararán las variables actuales con ellas mismas en lugar de hacerlo con las del registro (permanente).

ii.vi Si se detecta servicio operativo, entonces se evalúa el mac count.
ii.vi.i Si el mac_count original y actual son iguales, variables "mac_count_more" y "mac_count_less" en False.
ii.vi.ii Si el actual es mayor al original entonces: mac_count_more = True y la otra False.
ii.vi.iii Si el actual es menor al original entonces: mac_count_less = True y la otra False.
ii.vii Si se detecta servicio operativo, entonces se buscan las macs originales del servicio (las del registro) en el barrido actual, con el fin de detectar cambios de mac, por medio de un doble ciclo.
ii.vii.i Mientras el contador 1 sea menor al mac_count original:
ii.vii.i.i Mientras el contador 2 sea menor al mac_count actual:
ii.vii.i.i.i Evalúa si el objeto del índice de la lista original es igual al del actual.
ii.vii.i.i.i.i Si es igual, variable de cambio en False y se sale del subciclo.
ii.vii.i.i.i.ii Si es distinta, variable de cambio en True e incrementa en uno el cont2 para comparar la misma mac del listado original con la del siguiente índice de la actual, si existe. Si no, se sale del subciclo.
ii.vii.i.ii Si variable de cambio es True:
ii.vi.i.ii.i Actualiza lista booleana de control de cambio en el subíndice especifico como True
ii.vi.i.ii.ii Si no, deja el índice en False.
ii.vii.i.iii Se reinicia el contador 2 y aumenta en uno el contador 1 para evaluar la siguiente MAC del listado original, si existe. Si no, se sale del ciclo.
ii.viii Se verifica los mensajes a desplegar de acuerdo a estado de banderas de diagnóstico:
ii.viii.i Si el servicio está caído, despliega un mensaje correspondiente y:
ii.viii.i.i Si hay menos MACs que interfaces activas, activa un mensaje.
ii.viii.i.ii Por cada interfaz de la que no provenga MAC, despliega un mensaje.
ii.viii.ii Si el servicio se detecta operativo:
ii.viii.ii.i Si hay menor cantidad de MACs de lo esperado, activa mensaje.
ii.viii.ii.ii. Si hay mayor cantidad de MACs de lo esperado activa bandera/mensaje y la variable booleana vpls_update se coloca en True.
ii.viii.ii.iii Se revisa lista booleana de cambios de MAC:
ii.viii.ii.iii.i Si se detecta un cambio de mac y el count es el mismo, activa bandera/mensaje específico más el cambio de MAC ocurrido, y vpls_update = False.
ii.viii.ii.iii.ii Si se detecta un cambio de mac y el count es distinto, únicamente despliega el cambio de MAC ocurrido (el mensaje de mac count ya se habría desplegado en incisos i. y ii. del b del paso 13.) y vpls_update = False.
ii.viii.ii.iv Si vpls_update=True, actualiza la información de las MACs de ese servicio.
ii.viii.iii Si el servicio se detecta operativo, sin cambios de MAC ni de cantidad, servicio OK.
ii.ix Se procede a revisar el siguiente servicio.

3. IMPLEMENTACIÓN DEL ALGORITMO. Se aplica el algoritmo de monitoreo a 50 servicios de esta categoría. Para cumplir con esta cantidad, se aplicó el programa a ambos equipos PE Centrales que cuentan con servicios VPLS. La programación se basó en los pasos del algoritmo para asegurar un diagnóstico de nivel como parte del sistema de monitoreo. Se utilizó el lenguaje Python haciendo uso de librerías jnpr.junos y lxml. A continuación, se describe el desarrollo del programa en mención.

Paso 1: Importación de librerías, inicio de sesión y aplicación de RPCs. Se importaron las librerías necesarias y se inició la sesión con el equipo a monitorear, tal y como se mostró en las figuras 6.38 y 6.39. Se obtuvo la información necesaria de tablas MAC vía RPC y se convirtió en formato STRING (como se mostró en la figura 6.42) el formato XML.

Paso 2: Parseo inicial de servicios VPLS por tablas MAC. Se obtuvo la cantidad de tablas MAC por medio del elemento tag, l2ald-mac-entry, y se separaron los servicios por medio de una lista. Es decir, una lista para tablas MAC en las que la información completa de cada tabla de cada servicio VPLS, representó algún elemento de la lista. Para determinar dicha cantidad, se utilizó el método count de STRING y el resultado se dividió en dos, sabiendo que por cada servicio hay un elemento tag de apertura y uno de cierre, evitando que se duplicaran la cantidades reales. Para la separación de los elementos/servicios, se utilizó el método split de STRING, definiendo el largo de cada lista en función de la cantidad determinada, multiplicada por dos. Esto tomando en cuenta que se utilizan dos elementos tag (uno de apertura y el otro de salida) por cada servicio VPLS, quedando en los índices pares información inservible, y en los impares, la información deseada por servicio. Si no se multiplica por dos, no se realizaría la correcta separación de todos los servicios.

Figura#115: Parseo inicial de servicios VPLS por tablas MAC

```
get_vpls_macs_rpc = dev.rpc.get_vpls_mac_table(logical_system = [redacted] #Se obtienen todas las MAC
get_vpls_macs_tostring = etree.tostring(get_vpls_macs_rpc) #El resultado del GET se convierte en string.
cantidad_vpls_mx960 = get_vpls_macs_tostring.count('l2ald-mac-entry')/2  #El tag element ayuda a determinar
vpls_service_mac_table_list = get_vpls_macs_tostring.split('l2ald-mac-entry', cantidad_vpls_mx960*int('2'))
```

Paso 3: Inicio de evaluación individual de tablas MAC y determinación de cantidad de MACs por servicio VPLS. Se realizó un ciclo general cuyo fin fue el análisis (parseo de la información) de las tablas MAC individualmente, para luego poder realizar el diagnóstico servicio por servicio, en función de la cantidad de tablas MAC (determinada en el paso 2) multiplicada por dos (por lo que ya se explicó en paso 2) y haciendo referencia sólo a los elementos impares de la lista, ya que ahí se ubica la información deseada de cada servicio. Se guardó en una variable la información de la tabla MAC del servicio VPLS de interés, ubicada en el elemento impar de la lista correspondiente, al cual se hacía referencia dependiendo del ciclo. Luego, se determinó la cantidad de MACs por servicio VPLS por medio del elemento tag l2-mac-address. Esto siguió la misma lógica aplicada en el paso 2.

Figura#116: Parseo de tablas MAC y determinación de cantidad de MACs por servicio VPLS

```
while (vpls_selector < cantidad_vpls_mx960*int('2')): #Se garantiza que el c
    if (vpls_selector%2 != 0): #Se sabe que los strings deseados estan en lo
        vpls_service_mac_table = vpls_service_mac_table_list[vpls_selector]
        mac_count = vpls_service_mac_table.count('l2-mac-address')/2 # Se de
        mac_count_string = str(mac_count)
        mac_count_current = mac_count     #Para evitar posibles errores en e
        vpls_selector2 = vpls_selector2+1 #Enumera los servicios VPLS, solo
```

Paso 4: Parseo de información general por tabla MAC de servicio VPLS. Luego, haciendo uso de los elementos tag *l2-mac-routing-instance, l2-mac-logical-system* y *l2-bridge-vlan,* se obtuvo información general del servicio VPLS en evaluación. Esta información fue el nombre de la VPLS

(instancia), del sistema lógico y el *vlan-id*. Adicional, se incluyó el número de MACs en dicha lista. El parseo de la información se realizó por medio del método *split*, indicando dividir la información en función del elemento tag respectivo. Se utilizó el tag *"elemento-tag>"*. No se utilizó el carácter inicial *"<"*, ya que es distinto entre un elemento tag de apertura (*"<"*) y uno de cierre (*"</"*). Como puede verse en la imagen, el análisis o parseo de la información involucró varios pasos. Ya que se sabe que solo existe una variable por cada uno de estos tres elementos tag para cada servicio (a diferencia de las direcciones MAC, que pueden ser varias por servicio), entonces se crearon listas iniciales de tres elementos (0, 1, 2). Se sabe que la información deseada está en el índice 1, por lo que se guardó en variables respectivas. Sin embargo aún fue necesario volver a hacer listas de tres elementos para cada variable con el fin de parsear el carácter *"<"*, que como ya se explicó no pudo ser tomado en cuenta desde en el inicio de este parseo. Se sabe ahora que los datos totalmente parseados están en los índices 0 de estas nuevas listas, se guardaron en variables, las cuales terminaron formando parte de la lista de información general por servicio VPLS.

Figura#117: Parseo de información general por servicio VPLS

```
'''*****************************************************************************
VPLS INFO: Se hacen una lista que contiene información general de VPLS
*****************************************************************************
'''
vpls_name_parse_list = vpls_service_mac_table.split('l2-mac-routing-instance>', 2)
logical_system_parse_list = vpls_service_mac_table.split('l2-mac-logical-system>',2)
vlan_id_parse_list = vpls_service_mac_table.split('l2-bridge-vlan>',2)
#Se obtiene el elemento deseado de la lista pero aun no esta completamente parseado
vpls_name_parse_element = vpls_name_parse_list[1]
logical_system_parse_element = logical_system_parse_list[1]
vlan_id_parse_element = vlan_id_parse_list[1]
#Se hace una lista para completar el parseo'''
vpls_name_parse_list2 = vpls_name_parse_element.split('<', 1)
logical_system_parse_list2 = logical_system_parse_element.split('<', 1)
vlan_id_parse_list2 = vlan_id_parse_element.split('<', 1)
#Se obtiene el elemento totalmente parseado como string'''
vpls_name = vpls_name_parse_list2[0]
logical_system = logical_system_parse_list2[0]
vlan_id = vlan_id_parse_list2[0]
#Lista con informacion general de la VPLS'''
vpls_info_list = [vpls_name, logical_system, vlan_id, mac_count_string]
```

Paso 5: Determinación de interfaces lógicas asociadas al servicio VPLS. Por medio del RPC correspondiente, se obtuvo la información de las conexiones de la VPLS en evaluación. La VPLS fue identificada gracias al nombre de la misma guardada previamente como parte de la información general del servicio en evaluación. Esa información se guardó en un *String, y* se utilizó el elemento tag *"label-block-heading"* para separarla en dos por medio de una lista. Se guardó en una variable el primer elemento, que fue el de interés. Esto se hizo con el fin de poder utilizar adecuadamente el elemento tag *"interface-name>*, dado que el mismo se repite para interfaces remotas de los sitios ya previamente identificadas antes del elemento tag *"label-block-heading"*, y así evitar redundancia no deseada. El elemento tag "*interface-name",* en la forma *"<elemento-tag>"*, se utilizó para determinar la cantidad de interfaces lógicas. El mismo elemento tag, en la forma *"elemento-tag>"*, (por lo explicado con anterioridad) se utilizó para hacer una lista y dividir cada interfaz en un elemento de la misma. Por medio de un ciclo, similar al general, en función de la cantidad de interfaces totales del servicio en turno (multiplicada por dos), se obtuvo la información de los elementos impares de la lista (las de interés). Se parseó de forma parecida a como se hizo en el paso 4, y se almacenaron en lista, conformando así, la lista de interfaces de conexiones de la VPLS. Se realizó otro ciclo en función de la cantidad de interfaces totales, y se determinaron las cantidades de interfaces locales y las cantidades de interfaces remotas. La suma de ambas, siempre fue la de la cantidad total de la conexión VPLS.

Figura#118 Determinación de interfaces lógicas por servicio VPLS

```
'''*****************************************************************************************************************************************
                                VPLS CONNECTIONS EXTENSIVE, determinar sitios remotos y sedes locales
    *****************************************************************************************************************************************
'''
#Se obtiene la informacion de VPLS connections de interes
get_vpls_connection_rpc = dev.rpc.get_vpls_connection_information(logical_system =[redacted], instance=vpls_name, extensive=True)
vpls_service_connections = etree.tostring(get_vpls_connection_rpc)
connect_count=0
vpls_connections_list = vpls_service_connections.split('<label-block heading', 1)
vpls_connections = vpls_connections_list[0]
connect_count = vpls_connections.count('<interface-name>')
#Lista de interfaces asociadas locales (para determinar cantidad de sedes) y remotas (para sitios remotos)
interfaces_parse_list= vpls_connections.split('interface-name>', connect_count*2)
count=count2=0
interfaces_list= [None] * connect_count
while (count < connect_count*2):
    if (count%2 != 0):
        interfaces_list0= interfaces_parse_list[count].split('<',1)
        interfaces_list[count2] = interfaces_list0[0]
        count2= count2+1
    count = count +1
count = count2=0
```

```
#Se determina la cantidad de interfaces locales y remotas (sitios totales=  1+interfaces remotas) (sedes locales = interfaces locales)
connect_LSI_count=connect_Local_count=0
while (count < connect_count):
    if(interfaces_list[count].startswith("lsi.") == True):
        connect_LSI_count=connect_LSI_count+1
    if(interfaces_list[count].startswith("ge-") == True or interfaces_list[count].startswith("xe-") == True):
        connect_Local_count=connect_Local_count+1
    count=count+1
count=0
```

Por último, se determinó por medio del comando de configuración de la VPLS (no en la forma RPC, sino utilizando DEVICE.CLI(), por lo que se explicó con anterioridad), si existió una interfaz local activa primaria, para ignorar las demás interfaces locales. Esto aplicó únicamente si la cantidad de dichas interfaces era mayor a 1. Se ejecutó por medio de DEVICE.CLI(), el comando de la configuración sobre el servicio actual en evaluación. Si el parámetro *"active-interface"* era encontrado, entonces la cantidad de interfaces locales se fijaba a 1; se realizaba el parseo de la interfaz primaria activa de la salida del comando de la configuración; y se ejecutaba un ciclo que eliminaba las interfaces locales de la lista de interfaces de las conexiones VPLS, agregando al final la interfaz local primaria activa parseada, y modificando la variable de cantidad del total de interfaces de las conexiones VPLS según correspondiera. Luego, las tres cantidades de interfaces (total, locales, remotas) fueron almacenadas en una nueva lista.

Figura#118b Determinación de interfaces lógicas locales activas

```
'''VPLS Configuration
'''
#Si hay mas de una interfaz local activa, se procede a revisar si esta configurada como primaria (1 sitio) o debe
#de interfaces como sedes individuales.
if (connect_Local_count>1):
    configuration_vpls= dev.cli("show configuration logical-systems[redacted]routing-instances "+vpls_name
    redundancia="active-interface"
    boolean= redundancia in configuration_vpls #Si existe el parametro es porque hay interfaz primaria, por ende
    if(boolean==True):
        connect_Local_count=1
        interfaz_prim0=configuration_vpls.split('active-interface primary ',1)[1]
        interfaz_prim=interfaz_prim0.split(';',1)[0]
        while (count < connect_count):
            if(interfaces_list[0].startswith("ge-") == True or interfaces_list[0].startswith("xe-") == True):
                interfaces_list.pop(0)
                connect_count=connect_count-1
            count=count+1
        count=0
        interfaces_list.insert(0, interfaz_prim)
        connect_count=connect_count+1
connect_count_string, connect_LSI_count_string, connect_Local_count_string = str(connect_count), str(connect_LSI_

vplsconnections_info_list = [connect_count_string, connect_Local_count_string, connect_LSI_count_string]
```

Paso 6: Parseo individual de direcciones MAC y sus interfaces lógicas por servicio VPLS. El siguiente paso fue realizar el parseo de las direcciones MAC detectadas y sus respectivas interfaces lógicas, por servicio en evaluación. Se generó una lista inicial de MAC y otra de interfaces por medio de los elementos tag, "*l2-mac-address"* y "*l2-mac-logical-interface",* respectivamente, y en la forma:

"elemento-tag>" sin el caracter *"<"* por la razón explicada anteriormente. Se realizó un ciclo similar, en función de la cantidad de MACs multiplicado por dos, y se obtuvo la información de interés, ubicada en los elementos impares de la lista. El parseo fue similar, y la información íntegra fue almacenada en listas independientes. Una para direcciones MAC y la otra para interfaces.

Figura#119: Parseo individual de direcciones MAC y sus interfaces lógicas por servicio VPLS

```
'''*****************************************************************************************************
VPLS MACS Y LOGICAL IFS: Se hacen listas iniciales de parseo, el mac_count es vital para este parseo
*****************************************************************************************************
'''
mac_table_parse_list= vpls_service_mac_table.split('l2-mac-address>', mac_count*2)
logical_if_parse_list = vpls_service_mac_table.split('l2-mac-logical-interface>', mac_count*2)
'''Ciclo de parseo y recoleccion de macs y logical ifs
'''
count=count2=0
mac_table_list = [None]*mac_count
logical_if_list= [None] * mac_count
while (count < mac_count*2):
    if (count%2 != 0): #Si count es impar (residuo de division entre 2) ya que se sabe que las macs es
        mac_list0= mac_table_parse_list[count].split('<',1) #Se completa el parseo y se asigna a una l
        logical_list0 = logical_if_parse_list[count].split('<',1)
        mac_table_list[count2] = mac_list0[0] #Se asigna a la lista definitiva en la posicion count2''
        logical_if_list[count2] = logical_list0[0]
        count2= count2+1 #Se incrementa el count solo cuando se obtienen las macs de la primer lista,
    count = count +1 #Este contador se incrementa cada ciclo'''
count = count2=0
'''*****************************************************************************************************
'''
```

Paso 7: Despliegue ordenado de la información analizada por servicio VPLS. Se realizó el despliegue de la información analizada (parseada) al usuario por servicio VPLS. Cabe mencionar nuevamente que el ciclo general parsea y diagnóstica servicio por servicio, mostrando al usuario resultados en orden, por cada uno. Se utilizaron concatenaciones entre STRINGS y el método JOIN para unir listas a las cadenas. El despliegue por servicio mostrado al usuario puede verse en la sección de resultados.

```
'''*****************************************************************************************************
VPLS OVERALL MESSAGE: Se despliega la informacion completa de la VPLS
*****************************************************************************************************
'''
vpls_info_message = " "+str(vpls_selector2)+ ") "+vpls_info_list[0]+" overview: \n   Sistema logico: "+vpls_info_list[1]+ ", Vlan-id:
vpls_macs_message = "\n   VPLS Mac Table:  "+' | '.join(mac_table_list)
vpls_ifs_message = "   VPLS Logical IF: "+'    |    '.join(logical_if_list) +"\n \n   Resultado del diagnostico: "
connections_info_message = "   Sitios totales: "+str(connect_LSI_count+1)+ " (Locales=1, Remotos="+vplsconnections_info_list[2]+")"
connections_sede= "   Interfaces locales: "+vplsconnections_info_list[1]
connections_interfaces_message = "   Interfaces: "+', '.join(interfaces_list)
vpls_overall_message = ""+vpls_info_message+"\n"+connections_info_message+"\n"+connections_sede+"\n"+connections_interfaces_message+"\
print "++++++++++++++++++++++++++++++++++++++++++++++++++"
print vpls_overall_message
'''*****************************************************************************************************
```

Figura#120: Despliegue ordenado de la información analizada por servicio VPLS

Paso 8: Diagnóstico 1: Revisión de la cantidad de direcciones MAC y de interfaces lógicas. El servicio se considera no operativo si la cantidad de direcciones MAC reconocidas del servicio es menor a la cantidad de interfaces lógicas locales y remotas identificadas en las conexiones VPLS. La bandera que indica menos cantidad de MACs que lo esperado se activa. Es posible que aunque existan más direcciones MAC que interfaces asociadas a las conexiones VPLS, no se reciba MAC de alguna interfaz, y por tanto el servicio se detecte caído. Por ello lo siguiente es que se realiza un doble ciclo, en el que cada elemento de la lista de interfaces de conexión, se compara con cada elemento de la lista de interfaces de la tabla MAC, detectando que se reciba MAC de cada una. Se creó una lista booleana de control del mismo tamaño que la lista de interfaces de conexiones VPLS, en la que por cada interfaz sí detectada, el índice correspondiente se colocaba en "*False*", y por cada no detectada, el índice correspondiente en *"True".* Luego, por medio de otro ciclo se barrió esa lista booleana, en el que con un elemento detectado como "*True"*, se consideraba servicio caído. Todos los elementos debían estar en *"False"* para detectar servicio operativo. Así si se esperaba, por ejemplo, recibir de 4 interfaces lógicas remotas (LSI) (4 sitios PE) y sólo se identifican direcciones

MAC de tres LSI, entonces el servicio se detecta caído. La variable booleana de servicio caído (*service_down*) se activaba o desactivaba acorde a los resultados.

Figura#121: Diagnóstico 1: Revisión de la cantidad de direcciones MAC y de interfaces lógicas

```
'''1. VPLS Down por mac_count menor connect_count: Si hay menos mac
'''
if (mac_count<connect_count):
    service_down = True
    mac_count_less = True
'''2. VPLS Down por LSI o Local: Si del paso 1, service_down=False,
'''
interfaces_boolean_lost_list = [None]*connect_count
while(count<connect_count):
    while(count2<mac_count):
        if(interfaces_list[count]==logical_if_list[count2]):
            interface_presence = True
            count2=mac_count
        else:
            interface_presence = False
            count2=count2+1
    if(interface_presence==True):
        interfaces_boolean_lost_list[count]="False"
    if(interface_presence==False):
       interfaces_boolean_lost_list[count]="True"
    count2=0
    count=count+1
count=0
''' 3. Resultado primer fase (pasos 1 y 2): Se determina si el serv
'''
while(count<connect_count):
    if (interfaces_boolean_lost_list[count]=="True"):
        service_down = True
        count=connect_count
    else:
        service_down = False
        count=count+1
count=0
```

Paso 9: Asignación de variables permanentes (registros) por servicio VPLS. Se asignó la información general (instancia, sistema lógico, vlan, *mac-count*) y de comparación (lista de direcciones MAC y de interfaces lógicas) que queda almacenada de forma permanente en variables específicas por cada servicio. Esto con el fin de realizar la segunda parte del diagnóstico, basada en cambios de direcciones MAC y de cantidad. Si el servicio se detecta operativo y no se ha realizado asignación previa de variables permanentes; o si es necesario actualizar la información, entonces se realiza dicha asignación, y la información de cantidad de direcciones MAC y listas de MAC e interfaces se asigna a variables de comparación en el ciclo general actual. Si el servicio se detecta operativo y la asignación de variables permanentes ya se ha realizado, entonces únicamente se hace la asignación a variables de comparación para el ciclo actual. Finalmente, si no se cumplen ninguna de estas dos condiciones o si no se logra identificar el servicio VPLS actual, se asignan los valores detectados en el ciclo actual a las variables de comparación, dejando la segunda parte del diagnóstico sin ningún efecto, como se verá a continuación.

Figura#122 Asignación de variables permanentes (registros) por servicio VPLS

```
if (vpls_info_list[0]=="TEST-VPLS-[redacted]): #1
    if((service_down==False and if[redacted]=0) or vpls_update==True):
        vpls[redacted], mac[redacted] if[redacted] overall_[redacted] = vpls_info_list, mac_table_list, logical_if_list, vpls_overall_message
        mac_count_current, ifs_current, macs_current = int(vpls_[redacted][3]), if_[redacted], mac_[redacted]
    elif(service_down==False and if[redacted]=0):
        mac_count_current, ifs_current, macs_current = int(vpls_[redacted][3]), if_[redacted] mac_[redacted]
elif (vpls_info_list[0]=="VPLS[redacted]):#2
    if((service_down==False and if[redacted]==0) or vpls_update==True):
        vpls[redacted], mac[redacted] if[redacted], overall_[redacted] = vpls_info_list, mac_table_list, logical_if_list,
        mac_count_current, ifs_current, macs_current = int(vpls_[redacted][3]), if_[redacted], mac_[redacted]
    elif(service_down==False and if_[redacted]=0):
        mac_count_current, ifs_current, macs_current = int(vpls_[redacted][3]), if_[redacted], mac_[redacted]
elif (vpls_info_list[0]=="VPLS[redacted]):#3
```

Paso 10: Diagnóstico 2.1: Comparación de cantidad de MACs y lista booleana de cambios. La segunda parte del diagnóstico busca evaluar cambios de direcciones y de cantidad de MACs si el servicio se detecta operativo con la primera parte del mismo. Se define una lista booleana con longitud igual a la cantidad de MACs identificada del registro permanente. Si ese registro aún no existe para el servicio en mención, la variable de comparación, *mac_count_current* se iguala a la variable actual del conteo de MAC del servicio, *mac_count*, lo cual ocurre desde el paso 9. El primer paso es comparar la cantidad de MACs identificada del barrido del ciclo actual, con la cantidad almacenada en el registro permanente. Con base en los resultados, las banderas implicadas se activan o desactivan.

Figura#123: Diagnóstico 2.1: Comparación de cantidad de MACs y lista booleana de cambios

```
mac_boolean_change_list = [None]*mac_count_current #
'''**************************************************
   5. Evaluacion del MAC Count: Si se detecta servic
**************************************************
'''
if (service_down==False):
    if(mac_count_current==mac_count):
        mac_count_more, mac_count_less=False, False
    if(mac_count>mac_count_current):
        mac_count_more,mac_count_less=True,False
    if(mac_count<mac_count_current):
        mac_count_more,mac_count_less=False,True
    '''**********************************************
    '''
```

Paso 11: Diagnóstico 2.2: Comparación de direcciones MAC permanentes y actuales. Si el servicio se detecta operativo, entonces se comparan las direcciones MAC del registro permanente con las del barrido actual por medio de un doble ciclo. Se comparó cada elemento de la lista de MAC permanentes con cada uno de los elementos de la lista de MAC del barrido actual, con el fin de encontrar esa MAC "permanente". Si para la MAC en turno del listado permanente se encontraba una coincidencia, entonces se actualizaba el índice en turno de la lista booleana de cambios con un "*False*", dando a entender que esa MAC se sigue identificando. Si para la MAC en turno del listado permanente no se encontraba finalmente una coincidencia, entonces el índice en turno de la lista booleana de cambios se le asignaba un *"True"*, dando a entender que esa MAC ya no se identificaba, y que por tanto había habido un cambio en la tabla MAC del servicio VPLS en evaluación.

Figura#124: Diagnóstico 2.2: Comparación de direcciones MAC permanentes y actuales

```
'''*****************************************************************
6. Evaluacion de Mac addresses: Si se detecta servicio operativo, ent
*****************************************************************
'''
count_origin= count_now =0
while(count_origin<mac_count_current):
    while(count_now<mac_count):
        if (macs_current[count_origin]==mac_table_list[count_now]):
            mac_address_change=False
            count_now=mac_count
        else:
            mac_address_change=True
            count_now=count_now+1
    if(mac_address_change==True):
        mac_boolean_change_list[count_origin]="True"
    if(mac_address_change==False):
        mac_boolean_change_list[count_origin]="False"
    count_now=0
    count_origin=count_origin+1
count_origin=0
'''*****************************************************************
```

Paso 12: Despliegue de mensajes de acuerdo a diagnóstico (1). De acuerdo a las dos partes del diagnóstico realizado, se muestran ciertos mensajes al usuario. Primero se definieron dos variables de mensaje, una para cambios de direcciones MAC y la otra para interfaces de las que no se

detectaran MAC. Luego, se inició con el proceso de despliegue de mensajes. Si el servicio se detectaba no operativo porque la cantidad de direcciones MAC es menor a la cantidad de interfaces lógicas activas, entonces se desplegó un mensaje que indica esto al usuario. Se hizo un ciclo en el que se evaluó cada índice de la lista booleana de interfaces, en donde para cada índice en *"True"*, se desplegaba un mensaje indicando la interfaz sin direcciones MAC asociadas. Si el servicio se detectaba no operativo sólo por no recibir direcciones MAC de más de alguna interfaz (es decir, la cantidad de direcciones MAC era mayor o igual que la cantidad de interfaces de las conexiones VPLS), se indicó esto por un mensaje y se realizó el mismo ciclo que en el caso anterior.

Figura#125: Despliegue de mensajes de acuerdo a diagnóstico (1)

```
*************************************************************************************************
   7. Mensajes a desplegar de acuerdo a diagnostico
*************************************************************************************************
'''
mac_address_change_message = "None"  #Se define asi para identificar si este cambio se da o no '''
interfaces_lost_message ="None"
'''7.1 Mensajes para servicio detectado como caido
'''
if (service_down==True):
    if (mac_count_less==True):
        print mac_count_less_than_interfaces_message
        print no_interfaces_message
        while(count_origin<connect_count):
            if(interfaces_boolean_lost_list[count_origin]=="True"):
                interfaces_lost_message = "         > Interfaz: "+interfaces_list[count_origin] +""
                print interfaces_lost_message
            count_origin = count_origin+1
        count_origin=0
    else:
        print no_interfaces_message0
        print no_interfaces_message
        while(count_origin<connect_count):
            if(interfaces_boolean_lost_list[count_origin]=="True"):
                interfaces_lost_message = "         > Interfaz: "+interfaces_list[count_origin] +""
                print interfaces_lost_message
            count_origin = count_origin+1
        count_origin=0
```

Paso 13: Despliegue de mensajes de acuerdo a diagnóstico (2). La segunda parte del despliegue consistió en indicar mensajes por servicio que se detectó operativo, pero con cambios de direcciones MAC y/o de cantidad. Si se detectan menos o más direcciones MAC que lo que indica el registro permanente, entonces se le indica al usuario que el servicio se ve operativo pero que debe considerar el cambio de cantidad. Si hay más, entonces la variable booleana de actualización de registros permanentes se activa, aunque debe cumplirse una condición posterior. Luego, mediante un ciclo se analiza cada elemento de la lista booleana de cambios de direcciones MAC. Si se detecta un cambio de MAC, y el número de direcciones se mantiene, entonces se despliega un mensaje que indica esto; la variable booleana de actualización de registros se desactiva. Si se detecta un cambio de MAC y la cantidad de direcciones cambia, únicamente se omite parte del mensaje a desplegar para evitar redundar (el mensaje de cambio de cantidad de MACs ya se habría desplegado con anterioridad) y la variable booleana de actualización se desactiva. Por último se verifica el estado de esta variable de actualización de registros. Como se puede deducir de lo mencionado, esta variable se mantiene activa únicamente si hay más direcciones MAC y todas las MAC originales se mantienen. Si se encuentra activa, entonces realiza nuevamente la asignación de variables permanentes (registros) como forma de actualización.

Figura#126: Despliegue de mensajes de acuerdo a diagnóstico (2)

```
    ''' 7.2 Mensajes para servicio que se detecta operativo pero con cambios en
    '''
    if (service_down == False):
        if (mac_count_less == True):
            print vpls_up_conditional + "\n"+mac_count_less_message
        if (mac_count_more == True):
            print vpls_up_conditional + "\n"+mac_count_more_message
            vpls_update= True  # Pero ademas de que hayan mas MACs tambien debe
        while(count_origin<mac_count_current):
            if(mac_boolean_change_list[count_origin]=="True"):
                if(mac_count_less==False and mac_count_more == False):
                    print vpls_up_conditional + "\n"+mac_count_same_message  #S
                vpls_update=False # Si falta una MAC la condicion para actualiz
                mac_address_change_message = "         > La direccion MAC: "+ma
                print mac_address_change_message
            count_origin = count_origin+1
        count_origin=0
        ''' Si hay mas macs y las originales se mantienen entonces actualiza la
        '''
        if (vpls_update==True):
            print vpls_update_message
            assign permanent variables()
```

Paso 14: Despliegue de mensajes de acuerdo a diagnóstico (3). Por último se despliega un mensaje si no se encuentran problemas de ninguna índole al realizar el diagnóstico. Para ello se utilizan variables booleanas que evalúan que el servicio se considere operativo, que no haya más ni menos direcciones MAC y que no haya cambios en las direcciones para cumplir con la condición de desplegar el mensaje. Luego se incrementa el selector para evaluar el siguiente servicio en el próximo ciclo.

Figura#127 Despliegue de mensajes de acuerdo a diagnóstico (3)

```
    ''' 7.3 Mensaje para servicio que se detecta operativo sin cambios de mac addresses ni mac count
    '''
    if (service_down == False and mac_count_more == False and mac_count_less == False and mac_address_change_message=="None"):
        print vpls_ok_message
    print mac_boolean_change_list #print con fin unico de control en la evaluacion de este programa
    print "++++++++++++++++++++++++++++++++++++++++++++++++++\n"
    '''************************************************************************************************************************************************
    '''
'''************************************************************************************************************************************************
'''
vpls_selector = vpls_selector+1 #Se procede a revisar el siguiente servicio'''
time.sleep(1)
'''************************************************************************************************************************************************
'''
```

4. PRUEBAS Y ANÁLISIS DEL ALGORITMO. Se determinó la eficacia del algoritmo implementado, haciendo pruebas hacia uno de los equipos PE centrales. Se probaron todos los escenarios posibles y se observó el comportamiento y los resultados. Al observar resultados no deseados en lo que se esperaba de cada escenario, se hacían las correcciones necesarias. A continuación, se resumen los escenarios que fueron probados. Los resultados esperados por cada uno, son mostrados en la sección de resultados.

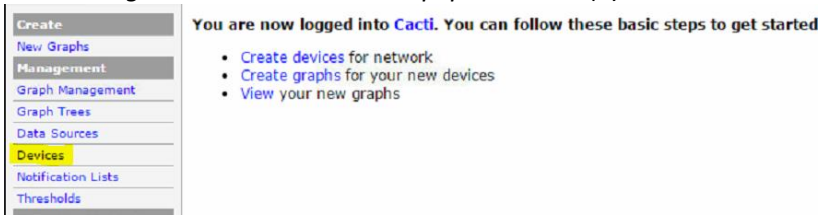
- Servicio VPLS punto-punto operativo.
- Servicio VPLS punto-multipunto operativo.
- Servicio VPLS con varias interfaces locales y una detectada como activa primaria.
- Servicio VPLS con varias interfaces locales y todas activas (no hay activa primaria).
- Servicio VPLS no operativo por tener menos direcciones MAC que interfaces lógicas activas.
- Servicio VPLS no operativo por no recibir direcciones MAC de interfaces lógicas remotas (LSI).
- Servicio VPLS no operativo por no recibir direcciones MAC de interfaces lógicas locales activas.
- Servicio VPLS operativo pero con menos direcciones MAC de las esperadas lo que implica cambios en direcciones MAC originales.
- Servicio VPLS operativo pero con más direcciones MAC de las esperadas y con cambios en direcciones MAC originales.
- Servicio VPLS operativo pero con más direcciones MAC de las esperadas y sin cambios en direcciones MAC originales (actualización de registros).
- Servicio VPLS con misma cantidad de direcciones MAC pero con cambios en direcciones MAC originales.

5. GRÁFICAS DE TRÁFICO: Una parte simple pero fundamental en el monitoreo de los servicios fue graficar el tráfico de los servicios. Simple en el sentido que Cacti provee los scripts y herramientas de poleo necesarias para generarlas, siempre y cuando se tenga lectura SNMP de los dispositivos de interés. Fundamental en el sentido que es el primer paso para determinar sin un servicio es candidato o no a estar no operativo. El tráfico se obtuvo de las interfaces de salida de los PE central y remotos implicados en el servicio. Esto debido a que para enlaces punto-multipunto graficar sólo del PE central no es suficiente. El problema se puede dar cuando para un PE remoto existan dos o más sedes (esto se trata en la discusión de resultados). Debido a que no se tiene gestión de los PE remotos, fue necesario como primer paso, solicitar lecturas SNMP y realizar el poleo respectivo del equipo.

A continuación se detallan los pasos realizados una vez se contó con lectura SNMP de los equipos PE.

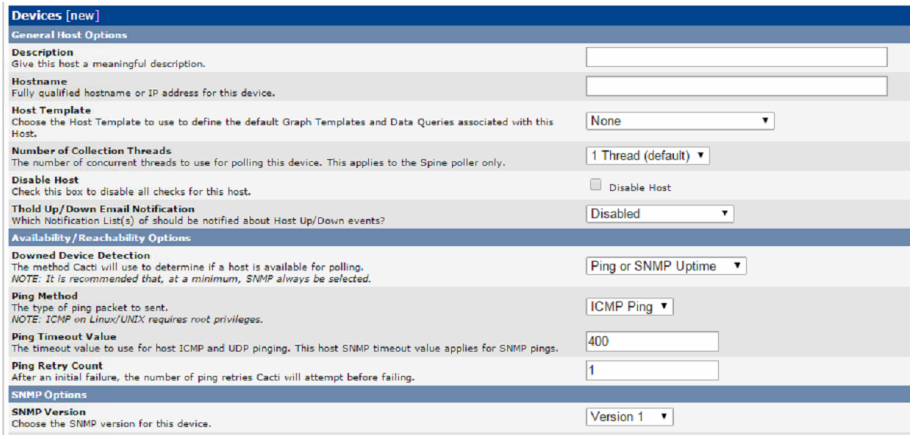
Paso 1: Poleo del equipo PE. El primer paso fue realizar el poleo del equipo para tener gestión sobre el mismo a través de Cacti. Para ello, se debió, desde la pestaña de inicio, ir a *"Devices"*. Luego, en la esquina superior derecha, presionar la opción *"Add"*.

Figura#128: Poleo de un equipo en Cacti (1)



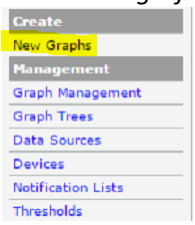
Luego se debió llenar la información requerida en la figura 6.63. La descripción de equipos es opcional pero sirve para identificarlos mejor. En el campo de Hostname se debe colocar la IP de gestión, y por último se debe indicar el uso de SNMP V1 (Versión 1). Al dar en la opción *"create"*, en la esquina inferior derecha, se crea el poleo del dispositivo.

Figura#129: Poleo de un equipo en Cacti (2)



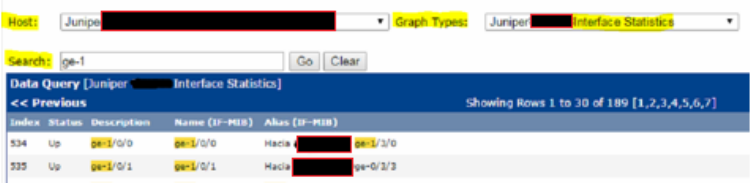
Paso 2: Generación de gráficas de tráfico. Desde la página inicial, en la esquina superior izquierda, se debe ir a la opción "New graphs".

Figura#130: Generación de gráficas de tráfico (1)



Se debe seleccionar el *"host"* requerido, el tipo de gráfica *"Interface statistics"* y especificar en "*search"*, la interfaz lógica que se desea graficar. Finalmente debe seleccionarse la opción y presionar en *"create"*.

Figura#131: Generación de gráficas de tráfico (2)



6. IDENTIFICACIÓN DE SUBINTERFACES Y VPLS. La identificación de subinterfaces y VPLS cumplió la misma función que la configuración de descripciones para servicios IP. Se debió identificar la interfaz en el PE central que llevara a la sede central del cliente, con respecto de otras posibles interfaces que estuvieran configuradas con otros fines, como por ejemplo de monitoreo o de conexión a otras sedes. Cabe destacar que para el PE central, fue bastante normal encontrar una única sede, la central. (Para referirse a cómo realizar esta identificación, ir a los pasos de monitoreo IP dentro de esta marco).

7. FORMA ALTERNATIVA DE MONITOREO: RPM SOBRE SERVICIOS VPLS. A pesar que para el suscriptor, los servicios VPLS se ofrecen de forma transparente (comunicación a nivel IP sólo es utilizada por el suscriptor), el proveedor puede realizar un monitoreo RPM con previa autorización y apoyo del suscriptor. Esto es usual en casos críticos y no resulta escalable. Se necesita que el suscriptor configure una red en sus sedes y reserve una dirección IP para el equipo de monitoreo del proveedor. A continuación, los pasos para el monitoreo, que resultan muy parecidos a los del monitoreo RPM para servicios IP. Se necesita de una conexión física entre el equipo de monitoreo y el equipo PE central.

a. Configuración de la subinterfaz del equipo de monitoreo: de tipo IPv4, la dirección IP corresponde a la dada por el suscriptor dentro de la red configurada por ellos.

Figura#132: Configuración de subinterfaz en equipo de monitoreo para RPM de VPLS

```
jrobles@jun.m10i[redacted]> show configuration interfaces ge-0/3/3.1315
description "Para routing instance de RPM para servicio VPLS";
vlan-id 1315;
family inet {
    address 10.200.10.6/29;
}
```

b. Configuración de la subinterfaz del equipo PE: de tipo VPLS, dado que la instancia de ruteo sobre la que se configura es de familia VPLS.

Figura#133: Configuración de subinterfaz de equipo PE para RPM de VPLS

```
jrobles@[redacted]MX960[redacted]> show configuration logical-systems [redacted] interfaces ge-1/0/1.1315
description "Servicio VPLS || Para monitoreo RPM";
encapsulation vlan-vpls;
vlan-id 1315;
family vpls;
```

c. Configuración de la instancia de ruteo en equipo de monitoreo: similar a la forma utilizada para servicios IP.

Figura#134: Configuración de la instancia de ruteo en equipo de monitoreo para RPM VPLS

```
jrobles@[redacted].m10i[redacted]-re0> show configuration routing-instances NOMBRE-VPLS_RPM
description "Para RPM de VPLS";
instance-type virtual-router;
interface ge-0/3/3.1315;
```

d. Configuración de la subinterfaz en VPLS de PE: la subinterfaz de tipo VPLS se incluye dentro de la VPLS configurada en el PE.

Figura#135: Configuración de subinterfaz en VPLS de PE

```
jrobles@[redacted]MX960[redacted]> show configuration logical-systems[redacted] routing-instances VPLS[redacted]
description "NOMBRE VPLS PAIS (73810)";
instance-type vpls;
interface ge-1/0/1.1315;
interface ge-1/0/2.1315;
```

e. Configuración de RPM en equipo de monitoreo: se observa en la siguiente imagen.

Figura#136: Configuración de RPM en equipo de monitoreo

```
jrobles@[redacted]m10i.[redacted]-re0> show configuration services rpm probe c43365-NOMBRE-VPLS
/* target[redacted] */
test c43365-BT_RPM-ping {
    target address 10.200.10.2;
    routing-instance NOMBRE-VPLS_RPM;
}
/* target:[redacted] */
test i73810-RPM-ping {
    target address 10.200.10.1;
    routing-instance NOMBRE-VPLS_RPM;
}
```

8. PLAN DE PRUEBAS DEL SISTEMA DE MONITOREO VPLS. Además de verificar el correcto funcionamiento técnico del algoritmo implementado, fue necesario realizar pruebas en escenarios reales con el fin de determinar su eficacia, combinando tanto la lectura de direcciones MAC dadas por el algoritmo como el comportamiento del tráfico, verificando así el sistema de monitoreo propuesto en su totalidad, llegando a conclusiones importantes para este Trabajo de Graduación.

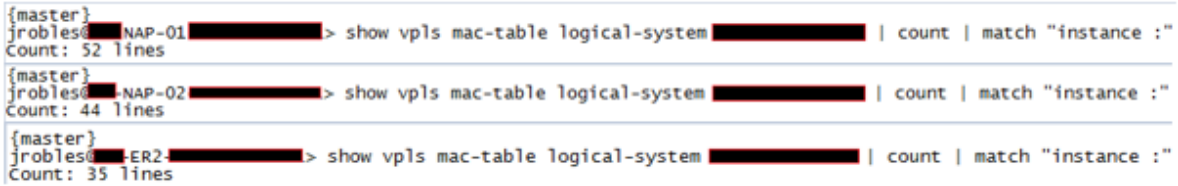
El sistema de monitoreo fue probado durante tres semanas y fue dividido en tres fases, una previa, una de planteamiento y una última de ejecución.

a. Fase previa: en esta etapa se realizaron las acciones necesarias para ejecutar las pruebas de monitoreo teniendo todos los recursos necesarios al alcance.

- Determinar los equipos bajo prueba. En este caso se escogieron tres equipos designados a partir de ahora como NAP1, NAP2 y ER.
- Del total de servicios VPLS encontrados, depurar los servicios VPLS que no estén en producción y con ello definir total de servicios bajo prueba en tres equipos. En este caso se implementó el sistema finalmente bajo 118 servicios (contrario a lo inicialmente planteado, ver sección VII. Análisis de Resultados).
- Obtener gráficas de tráfico de los servicios implicados.
- Modificar el programa de detección y diagnóstico de VPLS.
    - Se adaptó y aplicó el programa a cada uno de los equipos implicados (NAP1, NAP2 y ER) del ISP.
    - Se ignoraron los servicios VPLS depurados.
    - Se definió el ciclo de ejecución cada 10 minutos de acuerdo a las pruebas de tiempo de vida de direcciones MAC (MAC Aging Time).
    - Ventana emergente (pop-up) que muestre las alarmas posibles a generarse en los servicios. (Las alarmas fueron definidas en la fase de planteamiento).
- Configurar y realizar acciones sobre una VPLS de pruebas.
    - Sobre tiempo de vida de direcciones MAC para llegar
    - Sobre correcta generación de alarmas (de acuerdo a los escenarios provocados).

En la siguiente figura se puede observar el total de servicios VPLS encontrados por sus tablas de direcciones MAC.

Figura#137: Cantidad de servicios VPLS identificados por sus tablas MAC



b. Fase de planteamiento: en esta etapa se detallaron los escenarios posibles de diagnóstico, las alarmas a generar de acuerdo a dichos escenarios, la forma de realizar las pruebas y las acciones a tomar de acuerdo a las alarmas.

- Escenarios posibles de diagnóstico
    - Escenario 1: Servicio VPLS operativo.
        - Servicio VPLS punto-punto operativo.
        - Servicio VPLS punto-multipunto operativo.
    - Escenario 2: Servicio VPLS no operativo.
        - Servicio VPLS no operativo por tener menos direcciones MAC que interfaces lógicas activas.
        - Servicio VPLS no operativo por no recibir direcciones MAC de interfaces lógicas remotas (LSI).
        - Servicio VPLS no operativo por no recibir direcciones MAC de interfaces lógicas locales activas.
    - Escenario 3: Servicio VPLS detectado operativo pero con menos direcciones MAC de lo esperado.
    - Escenario 4: Servicio VPLS detectado operativo con misma cantidad de direcciones MAC pero con cambios en las mismas.
    - Escenario 5: Servicio VPLS detectado operativo pero con más direcciones MAC de las esperadas y con cambios en direcciones MAC originales.
    - Escenario 6: Servicio VPLS detectado operativo pero con más direcciones MAC de las esperadas y sin cambios en direcciones MAC originales (actualización de registros).

- Alarmas de acuerdo a escenarios
    - Sin alarmas: Escenario 1.
    - Alarma roja: Escenario 2.
    - Alarma naranja: Escenario 3 y 4.
    - Alarma amarilla: Escenario 5.
    - Alarma verde: Escenario 6.

- Forma de realizar las pruebas
    - Monitoreo de VPLS automatizado cada 10 minutos definiendo de 5 a 7 horas por día, 5 veces por semana, durante 3 semanas.
    - Llevar un registro diario, semanal y global.
    - Anotar cada evento que genere cualquier tipo de alarma, ordenado de acuerdo a tipo de alarma, escenario y servicio afectado.

- Llevar dentro del registro diario resumen de acciones aplicadas (reporte a operaciones locales implicadas del ISP, tipo de retroalimentación a cliente, caídas reales, en falso, desfase entre tiempo de detección de proveedor con respecto a reporte de cliente, etc).

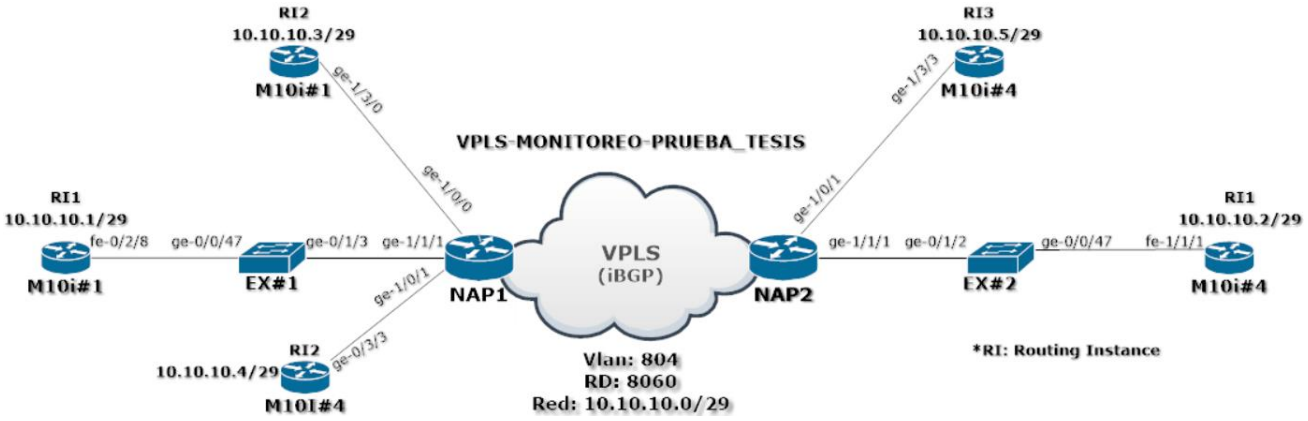
- Acciones generales a tomar de acuerdo a alarmas
  - Alarma roja
    - Anotar evento en el registro.
    - Monitoreo en tiempo real de tráfico y RPM (si aplica).
    - Confirmar a nivel de equipos que se haya perdido dirección(es) MAC especificada(s) por el programa de detección y diagnóstico.
    - Generar un ticket si no se ve MAC ni tráfico a nivel de proveedor.
    - Escalar a las operaciones implicadas para confirmar la caída del servicio.
    - Retroalimentar al cliente.
    - Llamar al cliente.
    - Llenar la tabla de control dentro del registro.
  - Alarma naranja:
    - Anotar evento el registro.
    - Monitoreo en tiempo real de tráfico, y RPM (si aplica).
    - Si se observa caída de tráfico, aplicar pasos de alarma roja.
    - Llenar la tabla de control dentro del registro.
  - Alarma amarilla:
    - Anotar el registro.
    - Monitoreo normal del tráfico, y RPM (si aplica).
    - Si se observa caída de tráfico, generar un ticket y aplicar pasos de alarma roja.
    - Llenar la tabla de control.
  - Alarma verde:
    - Anotar el registro.
    - Llenar la tabla de control.

a. Fase de ejecución: Se implementó el sistema de monitoreo sobre los servicios VPLS activos, realizando las acciones planteadas en la fase de planteamiento y haciendo uso de los recursos definidos en la fase previa. Adicionalmente, se determinó la forma de retroalimentar al suscriptor, dependiendo de la semana de ejecución.

- Semana 1: Generar caso (ticket) y escalar a la operación local del ISP respectiva.
- Semana 2: Generar caso (ticket), escalar a operación local y retroalimentar vía correo electrónico a suscriptor.
- Semana 3: Generar caso (ticket), escalar a operación local y retroalimentar vía correo electrónico a suscriptor.

9. VPLS DE PRUEBAS. Se configuró una VPLS de dos sitios (NAP1 y NAP2), definiendo tres interfaces locales para el sitio NAP1 y dos interfaces locales para el sitio NAP2. El diagrama de la red planteada puede verse a continuación. El fin fue utilizar esta VPLS para pruebas técnicas del programa de detección y diagnóstico elaborado, y como parte de las pruebas de tiempo de vida de direcciones MAC (MAC Aging Time).

Figura#138: Diagrama de VPLS de pruebas



Se ejecutaron las siguientes acciones durante la etapa de la configuración de la VPLS de pruebas:

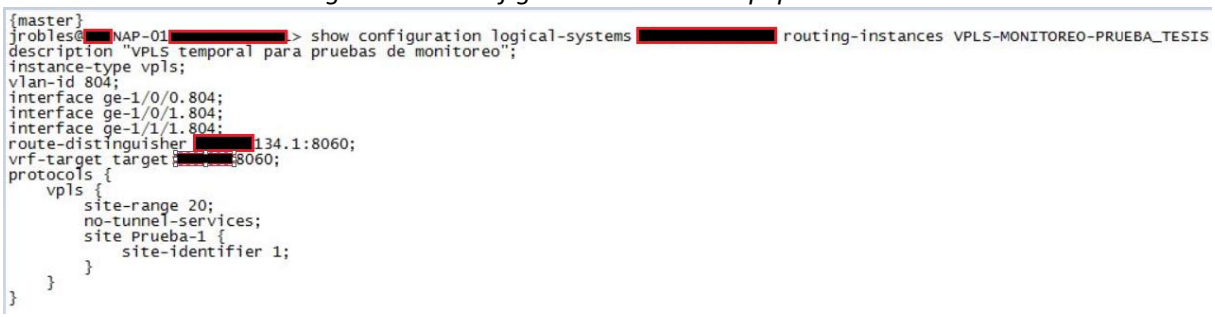
Se definió utilizar los equipos NAP1 y NAP2 para configurar la VPLS. Los mismos representaron los sitios remotos de una VPLS convencional. Se utilizaron los equipos de monitoreo, designados a partir de este momento como M10i#1 y M10i#2, para representar las sedes remotas (de las cuales se levantó un segmento IP), haciendo uso de las conexiones físicas existentes entre ellos. Se utilizó adicionalmente equipos de capa 2 (EX#1 y EX#2), como puede verse en la Figura 138, para una sede remota por cada sitio. Estos equipos, EX#1 y EX#2 representaron las redes de operaciones locales del ISP. Un resumen de los comandos (set commands) para todos los equipos puede verse en la sección de anexos.

Se verificó que la VLAN utilizada (804) no se estuviera siendo empleada en ninguno de los equipos. Se hizo lo mismo con el Route Distinguisher (8060). El resto de configuración de la VPLS en cada equipo fue el típico para dos equipos que tienen una sesión BGP establecida y de acuerdo a las configuraciones de los servicios VPLS convencionales del ISP.

- No tunnel services.
- Site Range: 20
- Sites identifiers: 1 (NAP1) y 5 (NAP2).

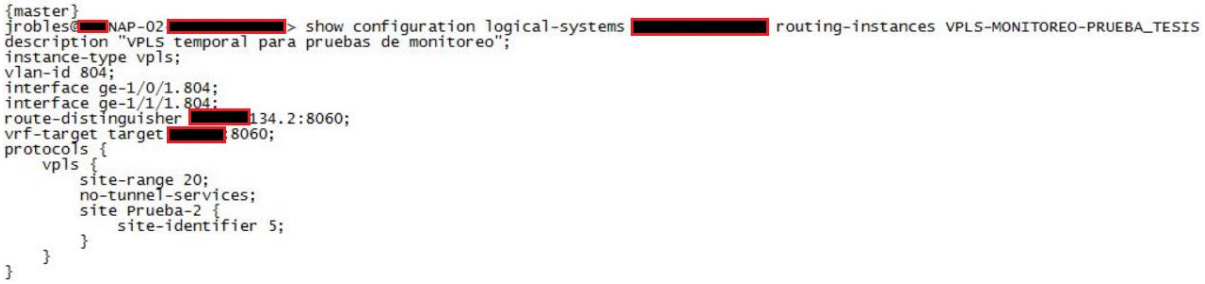
A continuación se observa la configuración de las VPLS de pruebas en NAP1 y NAP2.

Figura#139: Configuración VPLS en equipo NAP1



```
{master}
jrobles@[redacted]NAP-01[redacted]> show configuration logical-systems [redacted] routing-instances VPLS-MONITOREO-PRUEBA_TESIS
description "VPLS temporal para pruebas de monitoreo";
instance-type vpls;
vlan-id 804;
interface ge-1/0/0.804;
interface ge-1/0/1.804;
interface ge-1/1/1.804;
route-distinguisher [redacted]134.1:8060;
vrf-target target[redacted]8060;
protocols {
    vpls {
        site-range 20;
        no-tunnel-services;
        site Prueba-1 {
            site-identifier 1;
        }
    }
}
```

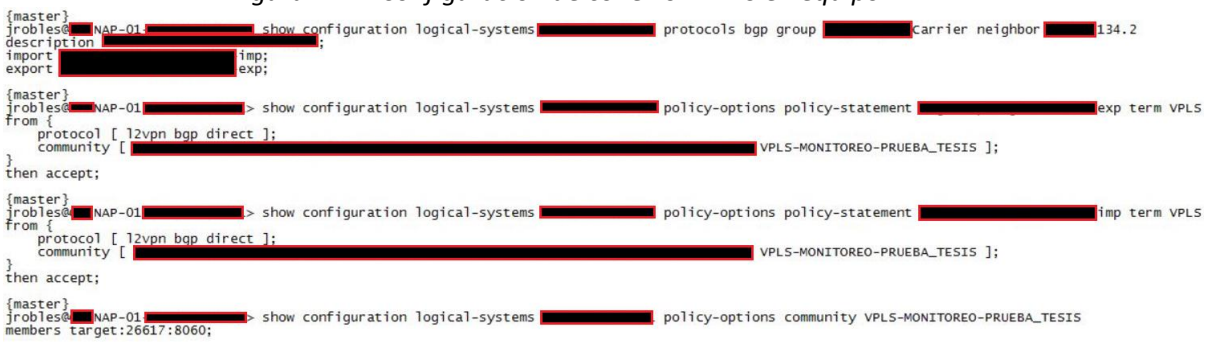
Figura#140: Configuración VPLS en equipo NAP2



```
{master}
jrobles@[redacted]NAP-02[redacted]> show configuration logical-systems [redacted] routing-instances VPLS-MONITOREO-PRUEBA_TESIS
description "VPLS temporal para pruebas de monitoreo";
instance-type vpls;
vlan-id 804;
interface ge-1/0/1.804;
interface ge-1/1/1.804;
route-distinguisher [redacted]134.2:8060;
vrf-target target[redacted]8060;
protocols {
    vpls {
        site-range 20;
        no-tunnel-services;
        site Prueba-2 {
            site-identifier 5;
        }
    }
}
```

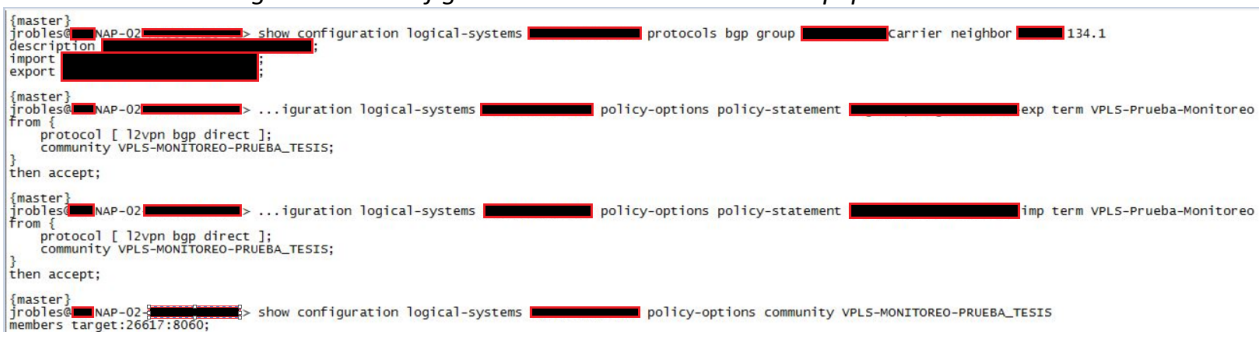
Se identificó la configuración de BGP entre los dos equipos. Se configuró la comunidad de la VPLS y se agregó adecuadamente a las políticas de importación y exportación aplicadas dentro de las configuraciones BGP. Esto se aplicó en ambos equipos (NAP1 y NAP2) como puede verse a continuación.

Figura#141: Configuración de conexión VPLS en equipo NAP1



```
{master}
jrobles@[redacted]NAP-01[redacted] show configuration logical-systems [redacted] protocols bgp group [redacted]Carrier neighbor [redacted]134.2
description [redacted];
import [redacted]imp;
export [redacted]exp;

{master}
jrobles@[redacted]NAP-01[redacted]> show configuration logical-systems [redacted] policy-options policy-statement [redacted]exp term VPLS
from {
    protocol [ l2vpn bgp direct ];
    community [ [redacted]VPLS-MONITOREO-PRUEBA_TESIS ];
}
then accept;

{master}
jrobles@[redacted]NAP-01[redacted]> show configuration logical-systems [redacted] policy-options policy-statement [redacted]imp term VPLS
from {
    protocol [ l2vpn bgp direct ];
    community [ [redacted]VPLS-MONITOREO-PRUEBA_TESIS ];
}
then accept;

{master}
jrobles@[redacted]NAP-01[redacted]> show configuration logical-systems [redacted] policy-options community VPLS-MONITOREO-PRUEBA_TESIS
members target:26617:8060;
```

Figura#142: Configuración de conexión VPLS en equipo NAP2



```
{master}
jrobles@[redacted]NAP-02[redacted]> show configuration logical-systems [redacted] protocols bgp group [redacted]Carrier neighbor [redacted]134.1
description [redacted];
import [redacted];
export [redacted];

{master}
jrobles@[redacted]NAP-02[redacted]> ...iguration logical-systems [redacted] policy-options policy-statement [redacted]exp term VPLS-Prueba-Monitoreo
from {
    protocol [ l2vpn bgp direct ];
    community VPLS-MONITOREO-PRUEBA_TESIS;
}
then accept;

{master}
jrobles@[redacted]NAP-02[redacted]> ...iguration logical-systems [redacted] policy-options policy-statement [redacted]imp term VPLS-Prueba-Monitoreo
from {
    protocol [ l2vpn bgp direct ];
    community VPLS-MONITOREO-PRUEBA_TESIS;
}
then accept;

{master}
jrobles@[redacted]NAP-02[redacted]> show configuration logical-systems [redacted] policy-options community VPLS-MONITOREO-PRUEBA_TESIS
members target:26617:8060;
```

Se agregó la red 10.10.10.0/29 en cada una de las subinterfaces de los equipos M10i#1 y M10i#4, como se ve en el diagrama. Estas subinterfaces, que representaron las sedes remotas, se agregaron a instancias de ruteo individuales. Por último se configuró RPM en ambos equipos bajo el estándar 1 para asegurar la presencia continua de direcciones MAC en la VPLS en ambos equipos (NAP1 y NAP2). Se implementaron gráficas de tráfico y RPM en Cacti para observar el comportamiento. Se realizó la configuración necesaria en los equipos EX#1 y EX#2 para pasar la VLAN a los equipos implicados en las subinterfaces correspondientes, de acuerdo a la Figura 138. (Un resumen de los comandos en todos los equipos implicados se puede ver en la sección de anexos).

Se verificó que cada una de las direcciones MAC recibidas en la VPLS de cada equipo correspondiera a las direcciones físicas de las interfaces implicadas en los equipos de monitoreo. Para ello fue necesario conocer las conexiones físicas de los equipos; en este caso, con el diagrama de la Figura 138 fue suficiente.

Se ejecutaron las siguientes acciones durante la etapa pruebas técnicas de la VPLS configurada.

- Escenario 1: Servicio normalizado (con 5 “sedes remotas”) previo al inicio. Durante la prueba, deshabilitar todas las subinterfaces de M10Is con IPs: 10.10.10.1/29, 10.10.10.3/29 y 10.10.10.4/29.
  Resultado esperado:
  - NAP1: Alarma Roja posterior a la prueba.
  - NAP2: Alarma Roja posterior a la prueba.

- Escenario 2: Servicio normalizado (con 5 “sedes remotas”) previo al inicio. Durante la prueba, deshabilitar subinterfaces M10Is 10.10.10.2/29 y 10.10.10.5/29.
  Resultado esperado:
  - NAP1: Alarma Roja posterior a la prueba.
  - NAP2: Alarma Roja posterior a la prueba.

- Escenario 3: Servicio normalizado (con 5 “sedes remotas”) previo al inicio. Durante la prueba, deshabilitar una única subinterfaz de M10Is entre 10.10.10.1/29, 10.10.10.3/29 ó 10.10.10.4/29.
  Resultado esperado:
  - NAP1: Alarma Roja posterior a la prueba.
  - NAP2: Alarma Naranja posterior a la prueba.

- Escenario 4: Servicio normalizado (con 5 “sedes remotas”) previo al inicio. Durante la prueba, deshabilitar una única subinterfaz M10Is entre 10.10.10.2/29 ó 10.10.10.5/29.
  Resultado esperado:
  - NAP1: Alarma Naranja posterior a la prueba.
  - NAP2: Alarma Roja posterior a la prueba.

- Escenario 5: Servicio con 4 “sedes remotas” deshabilitando una subinterfaz de M10is entre 10.10.10.1/29, 10.10.10.3/29 ó 10.10.10.4/29 previo al inicio. Durante la prueba, habilitar la subinterfaz deshabilitada y deshabilitar alguna de ellas que estuviera habilitadas al inicio. Por ejemplo: Deshabilitar 10.10.10.1/29 previo al inicio, y durante la prueba habilitarla pero al mismo tiempo deshabilitar la 10.10.10.3/29.
  Resultado esperado:

  - NAP1: Alarma Roja previo y posterior a la prueba.
  - NAP2: Alarma Naranja posterior a la prueba.

- Escenario 6: Servicio con 4 “sedes remotas” deshabilitando una subinterfaz de M10Is entre 10.10.10.2/29 ó 10.10.10.5/29 previo al inicio. Durante la prueba, habilitar la subinterfaz deshabilitada y deshabilitar alguna de ellas que estuviera habilitadas al inicio. Por ejemplo: Deshabilitar 10.10.10.2/29 previo al inicio, y durante la prueba habilitarla pero al mismo tiempo deshabilitar la 10.10.10.5/29.
  Resultado esperado:
  - NAP1: Alarma Naranja posterior a la prueba.
  - NAP2: Alarma Roja previo y posterior a la prueba.

- Escenario 7: Servicio con 3 “sedes remotas” deshabilitando dos subinterfaces de M10is entre 10.10.10.1/29, 10.10.10.3/29 ó 10.10.10.4/29 previo al inicio. Durante la prueba, habilitar las dos subinterfaces deshabilitadas y deshabilitar la que estuviera habilitada al inicio. Por ejemplo: Deshabilitar 10.10.10.1/29 y 10.10.10.3/29 previo al inicio, y durante la prueba habilitarlas pero al mismo tiempo deshabilitar la 10.10.10.4/29.
  Resultado esperado:
  - NAP1: Alarma roja previo y posterior a la prueba.
  - NAP2: Alarma amarilla posterior a la prueba.

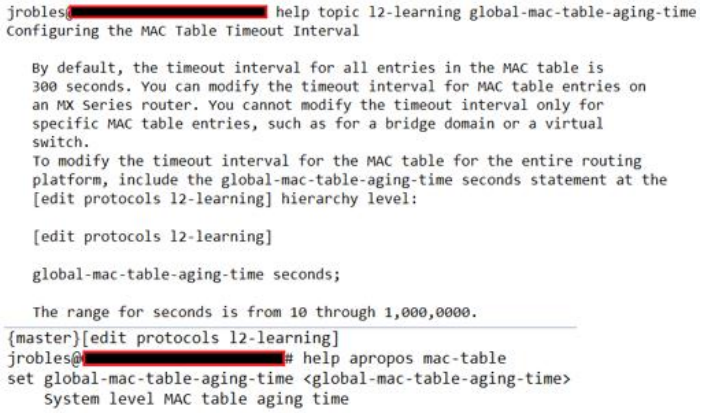
- Escenario 8: Servicio con 4 “sedes remotas” deshabilitando una subinterfaz de M10is entre 10.10.10.1/29, 10.10.10.3/29 ó 10.10.10.4/29 previo al inicio. Durante la prueba, únicamente habilitarla. Por ejemplo: Deshabilitar 10.10.10.1/29 previo al inicio, y durante la prueba habilitarla.
  Resultado esperado:
  - NAP1: Alarma roja previo a la prueba.
  - NAP2: Alarma verde posterior a la prueba.

- Escenario 9: Servicio con 4 “sedes remotas” deshabilitando una subinterfaz de M10is entre 10.10.10.2/29 ó 10.10.10.5/29 previo al inicio. Durante la prueba, únicamente habilitarla. Por ejemplo: Deshabilitar 10.10.10.2/29 previo al inicio, y durante la prueba habilitarla.
  Resultado esperado:
  - NAP1: Alarma verde posterior a la prueba
  - NAP2: Alarma roja previo a la prueba.

10. TIEMPO DE VIDA DE DIRECCIONES MAC EN UNA VPLS. Se verifico el tiempo de vida de direcciones MAC teórico, cómo configurarlo en los equipos NAP1, NAP2 y ER (en este caso todos pertenecen a la familia MX de Juniper) y la configuración que presentaron los equipos con respecto a esto.

En la imagen a continuación se puede ver el tiempo de vida de direcciones MAC teórico y cómo configurarlo en equipos de familia MX.

Figura#143: Tiempo de vida de direcciones MAC y cómo configurarlo en equipo familia Juniper MX



A continuación, se observa que ninguno de los equipos NAP1, NAP2, ER, presentan configuración de tiempo de vida de direcciones MAC, asumiendo el tiempo teórico (5 minutos, ver Figura 143).

Figura#144: Configuración de tiempo de vida de direcciones MAC en equipos a monitorear

Se realizaron pruebas de tiempo de vida de direcciones MAC sobre dos servicios VPLS con RPM hacia distintos equipos remotos. Se deshabilitaron las conexiones de los RPM bajo dos escenarios distintos y se contabilizó el tiempo de pérdida de las direcciones MAC respectivas tanto a nivel local como remoto. Adicionalmente se utilizó la VPLS de pruebas para este fin.

- Escenarios de pruebas sobre servicios VPLS.
  - Escenario 1: Configuración de Active-Interface-Primary sobre la subinterfaz del servicio para deshabilitar el resto de subinterfaces (las de RPM).
  - Escenario 2: Deshabilitar las subinterfaces del RPM mediante el comando “deactivate”.

- Escenarios de pruebas sobre VPLS de pruebas.
  - Escenario 1.a: Deshabilitar subinterfaz M10i#1.
  - Escenario 1.b: Deshabilitar subinterfaz NAP1.

- Escenario 2.a: Deshabilitar subinterfaz M10i#4.
- Escenario 2.b: Deshabilitar subinterfaz NAP2.
- Escenario 3: Deshabilitar subinterfaces M10i#1 y M10i#4.

VII. RESULTADOS

En este apartado se muestran los resultados obtenidos de los pasos desarrollados y documentados en el diseño experimental.

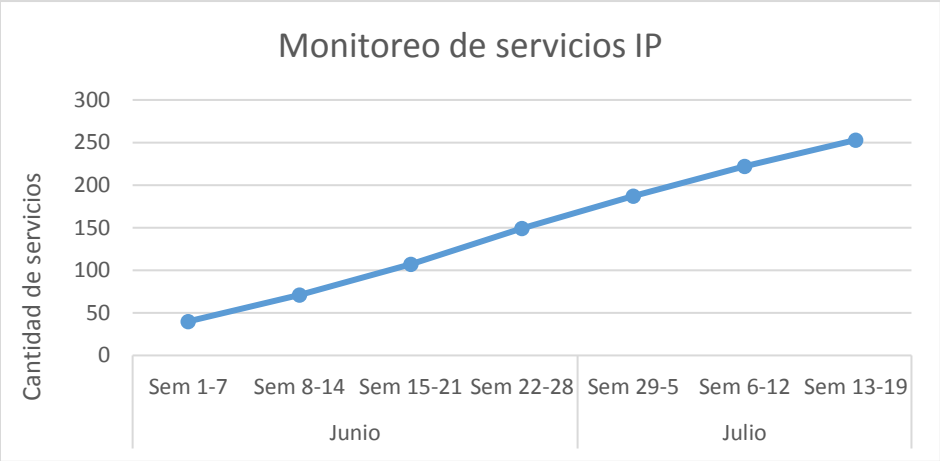
A. RESULTADOS DEL SISTEMA DE MONITOREO MPLS IP-VPN

La Figura#67 ilustra una representación gráfica de alto nivel del monitoreo de capa 3. Detalles sobre la misma se discuten en el análisis de resultados.

1. INVESTIGACIÓN Y CANTIDAD DE SERVICIOS IP MONITOREADOS. En el diseño experimental, se detallan distintos pasos para determinar las direcciones IP de los CPE de los servicios. No existe un orden determinado para ello y hay formas más eficaces que otras, pero según el caso o servicio, no siempre están disponibles, y se debe hacer uso de los otros recursos para asegurar un monitoreo correcto. La conectividad vía ping resultó obligatoria. El análisis de estos pasos y recursos de investigación de servicios IP se detalla en al análisis de resultados.

La forma de garantizar el monitoreo de todos estos servicios fue mediante un archivo de Excel importado del registro de la empresa que contiene todas las instalaciones activas. Se llevó un archivo de control de los servicios monitoreados con todos los detalles pertinentes para poder identificar fácilmente el monitoreo implementado. Se monitorearon poco más de 200 servicios y la gráfica de instalaciones monitoreadas en función del tiempo se muestra a continuación.

Figura#145: Avance de monitoreo de servicios IP



2. APLICACIÓN DEL SERVICIO RPM. A continuación, se detallan los resultados obtenidos en los pasos para la implementación del servicio RPM.

a. Sobre el ruteo entre equipo de monitoreo y PE central. Fue necesario establecer una conexión lógica entre el equipo de monitoreo y el equipo PE central, de tal forma de asegurar la conectividad vía ping hacia los puntos remotos de los servicios IP. A continuación, se muestran los resultados de este ruteo aplicado para servicios que pasan por una VRF del PE central.

En esta figura, se muestra la configuración de una VRF del PE central por la que pasa un servicio IP.

Figura#146: Configuración de una VRF en PE central

```
jrobles@[redacted]MX960-RE0[redacted]> show configuration logical-systems [redacted] routing-instances [redacted]
description "[redacted] (66673) - [redacted]";
instance-type vrf;
interface ge-1/0/0.12;
interface ge-1/0/1.1011;
route-distinguisher [redacted];
vrf-target target:[redacted];
vrf-table-label;
protocols {
    bgp {
        group [redacted] {
            type external;
            neighbor 68.136.228.30 {
                peer-as 1684;
            }
        }
    }
}
```

A continuación, la configuración de la sonda RPM para el servicio que pasa por la VRF en cuestión.

Figura#147: Configuración de sonda RPM para una VRF específica

```
jrobles@jun.m10i[redacted]-re0> show configuration services rpm probe c[redacted]-[redacted]
/* target:[redacted] */
test c[redacted]_RPM-ping {
    target address 68.136.228.30;
    routing-instance [redacted]-RPM;
}
/* target: [redacted] */
test i66673-RPM-ping {
    /* CPE; Guatemala */
    target address 172.23.1.253;
    routing-instance [redacted]-RPM;
}
```

Se muestra la efectividad del ping hacia los puntos remotos del servicio IP que pasa por esta VRF a través del ruteo realizado.

Figura#148: Conectividad vía ping hacia las sedes central y remota del servicio IP

```
jrobles@jun.m10i[redacted]> ping 68.136.228.30 routing-instance [redacted]-RPM rapid count 100
PING 68.136.228.30 (68.136.228.30): 56 data bytes
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
--- 68.136.228.30 ping statistics ---
100 packets transmitted, 100 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.544/0.604/0.824/0.041 ms

jrobles@jun.m10i[redacted]-re0> ping 172.23.1.253 routing-instance [redacted]RPM rapid count 100
PING 172.23.1.253 (172.23.1.253): 56 data bytes
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
--- 172.23.1.253 ping statistics ---
100 packets transmitted, 100 packets received, 0% packet loss
round-trip min/avg/max/stddev = 26.331/27.185/46.404/2.887 ms
```

En este caso particular, el telnet es permitido y de esta forma se garantiza que se trata del CPE en el punto remoto. Esto se realizó desde el equipo de monitoreo como se puede observar.

Figura#149: Conectividad vía Telnet hacia el CPE de un servicio IP

```
jrobles@jun.m10i.[redacted]-re0> telnet 172.23.1.253 routing-instance [redacted]RPM
Trying 172.23.1.253...
Connected to 172.23.1.253.
Escape character is '^]'.
CCCC
This system is to be used by authorized [redacted]Business Network Engineering
personnel only.  Individuals using this system without authority,
or in excess of their authority, are subject to having all activities
monitored and recorded by system personnel.
```

b. Sobre el funcionamiento de las sondas RPM. La configuración de las sondas RPM cumplió con tener un estándar definido. El estándar varió de acuerdo a la criticidad del servicio en tres categorías distintas (ver diseño experimental) diferenciadas por la frecuencia de sondeo (pruebas de ping y de RPM).

A continuación se muestran los resultados de sondas (*probe-results*) e históricos (*history-results)* para pruebas RPM (ping hacia las sedes remotas de servicios IP identificados por instalación) configuradas dentro de sondas RPM específicas (para las VRF de los PE centrales por los que pasan los servicios IP). Los mismos se dividen de acuerdo a los estándares aplicados según la criticidad.

- Resultados de sondas RPM con estándar 1

Figura#150: Resultados históricos de una prueba RPM con estándar 1

```
jrobles@[redacted] m10i[redacted]re0> show services rpm history-results owner c1234-NOMBRE-VRF test i53828-RPM-ping
    Owner, Test                 Probe received             Round trip time
    c1234-NOMBRE-VRF, i53828-RPM-ping Sun Sep 13 18:28:53 2015           66884 usec
    c1234-NOMBRE-VRF, i53828-RPM-ping Sun Sep 13 18:29:23 2015           51384 usec
    c1234-NOMBRE-VRF, i53828-RPM-ping Sun Sep 13 18:29:53 2015           51393 usec
    c1234-NOMBRE-VRF, i53828-RPM-ping Sun Sep 13 18:30:23 2015           51376 usec
    c1234-NOMBRE-VRF, i53828-RPM-ping Sun Sep 13 18:30:53 2015           54283 usec
    c1234-NOMBRE-VRF, i53828-RPM-ping Sun Sep 13 18:31:24 2015           51363 usec
```

Figura#151: Resultados de sonda de una prueba RPM con estándar 1

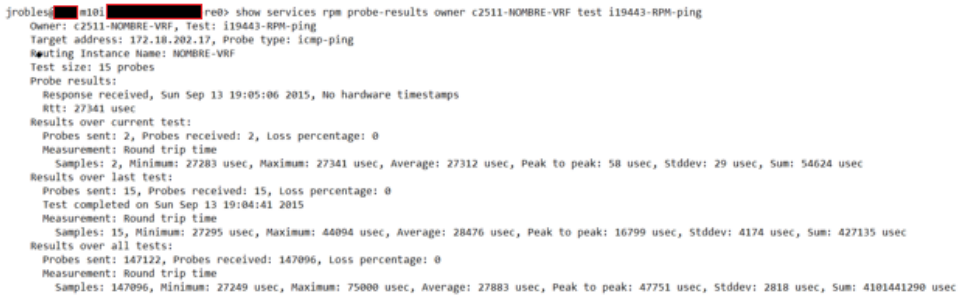
```
jrobles@[redacted]m10i[redacted] show services rpm probe-results owner c1234-NOMBRE-VRF test i53828-RPM-ping
    Owner: c1234-NOMBRE-VRF, Test: i53828-RPM-ping
    Target address: 193.155.246.198, Probe type: icmp-ping
    Routing Instance Name: NOMBRE-VRF
    Test size: 15 probes
    Probe results:
      Response received, Sun Sep 13 19:02:01 2015, No hardware timestamps
      Rtt: 51339 usec
    Results over current test:
      Probes sent: 12, Probes received: 12, Loss percentage: 0
      Measurement: Round trip time
        Samples: 12, Minimum: 51273 usec, Maximum: 51465 usec, Average: 51369 usec, Peak to peak: 192 usec, Stddev: 57 usec, Sum: 616433 usec
    Results over last test:
      Probes sent: 15, Probes received: 15, Loss percentage: 0
      Test completed on Sun Sep 13 18:53:59 2015
      Measurement: Round trip time
        Samples: 15, Minimum: 51321 usec, Maximum: 51771 usec, Average: 51432 usec, Peak to peak: 450 usec, Stddev: 129 usec, Sum: 771476 usec
    Results over all tests:
      Probes sent: 29532, Probes received: 29507, Loss percentage: 0
      Measurement: Round trip time
        Samples: 29507, Minimum: 39749 usec, Maximum: 824946 usec, Average: 48167 usec, Peak to peak: 785197 usec, Stddev: 10709 usec, Sum: 1421274369 usec
```

- Resultados de sondas RPM con estándar 2

Figura#152: Resultados históricos de una prueba RPM con estándar 2

```
jrobles@[redacted].m10i[redacted]-re0> show services rpm history-results owner c2511-NOMBRE-VRF test i19443-RPM-ping
    Owner, Test                 Probe received             Round trip time
    c2511-NOMBRE-VRF, i19443-RPM-ping Sun Sep 13 18:56:43 2015           27347 usec
    c2511-NOMBRE-VRF, i19443-RPM-ping Sun Sep 13 18:56:53 2015           27409 usec
    c2511-NOMBRE-VRF, i19443-RPM-ping Sun Sep 13 18:57:08 2015           32314 usec
    c2511-NOMBRE-VRF, i19443-RPM-ping Sun Sep 13 18:57:19 2015           27282 usec
```

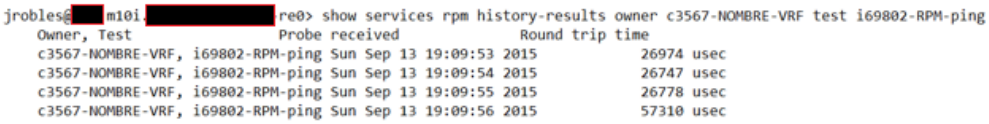
Figura#153: Resultados de sonda de una prueba RPM con estándar 2



```
jrobles@[redacted]m10i[redacted]re0> show services rpm probe-results owner c2511-NOMBRE-VRF test i19443-RPM-ping
    Owner: c2511-NOMBRE-VRF, Test: i19443-RPM-ping
    Target address: 172.18.202.17, Probe type: icmp-ping
    Routing Instance Name: NOMBRE-VRF
    Test size: 15 probes
    Probe results:
      Response received, Sun Sep 13 19:05:06 2015, No hardware timestamps
      Rtt: 27341 usec
    Results over current test:
      Probes sent: 2, Probes received: 2, Loss percentage: 0
      Measurement: Round trip time
        Samples: 2, Minimum: 27283 usec, Maximum: 27341 usec, Average: 27312 usec, Peak to peak: 58 usec, Stddev: 29 usec, Sum: 54624 usec
    Results over last test:
      Probes sent: 15, Probes received: 15, Loss percentage: 0
      Test completed on Sun Sep 13 19:04:41 2015
      Measurement: Round trip time
        Samples: 15, Minimum: 27295 usec, Maximum: 44094 usec, Average: 28476 usec, Peak to peak: 16799 usec, Stddev: 4174 usec, Sum: 427135 usec
    Results over all tests:
      Probes sent: 147122, Probes received: 147096, Loss percentage: 0
      Measurement: Round trip time
        Samples: 147096, Minimum: 27249 usec, Maximum: 75000 usec, Average: 27883 usec, Peak to peak: 47751 usec, Stddev: 2818 usec, Sum: 4101441290 usec
```

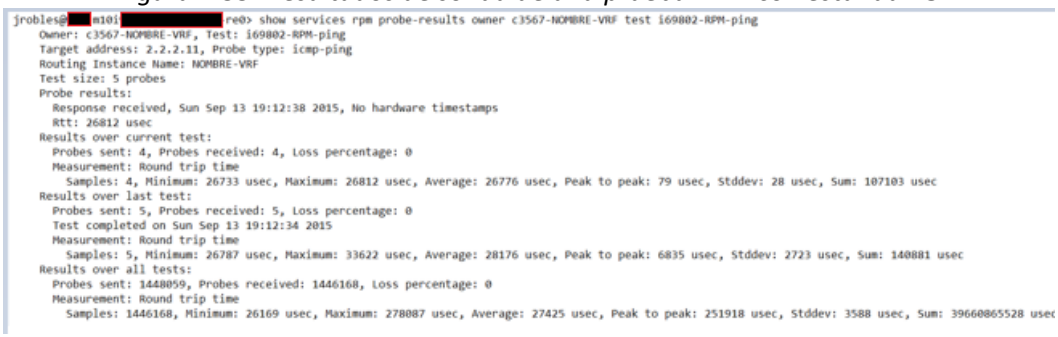
- Resultados de sondas RPM con estándar 3

Figura#154: Resultados históricos de una prueba RPM con estándar 3



```
jrobles@[redacted]m10i.[redacted]re0> show services rpm history-results owner c3567-NOMBRE-VRF test i69802-RPM-ping
    Owner, Test                  Probe received                Round trip time
    c3567-NOMBRE-VRF, i69802-RPM-ping Sun Sep 13 19:09:53 2015            26974 usec
    c3567-NOMBRE-VRF, i69802-RPM-ping Sun Sep 13 19:09:54 2015            26747 usec
    c3567-NOMBRE-VRF, i69802-RPM-ping Sun Sep 13 19:09:55 2015            26778 usec
    c3567-NOMBRE-VRF, i69802-RPM-ping Sun Sep 13 19:09:56 2015            57310 usec
```

Figura#155: Resultados de sonda de una prueba RPM con estándar 3



```
jrobles@[redacted]m10i[redacted]re0> show services rpm probe-results owner c3567-NOMBRE-VRF test i69802-RPM-ping
    Owner: c3567-NOMBRE-VRF, Test: i69802-RPM-ping
    Target address: 2.2.2.11, Probe type: icmp-ping
    Routing Instance Name: NOMBRE-VRF
    Test size: 5 probes
    Probe results:
      Response received, Sun Sep 13 19:12:38 2015, No hardware timestamps
      Rtt: 26812 usec
    Results over current test:
      Probes sent: 4, Probes received: 4, Loss percentage: 0
      Measurement: Round trip time
        Samples: 4, Minimum: 26733 usec, Maximum: 26812 usec, Average: 26776 usec, Peak to peak: 79 usec, Stddev: 28 usec, Sum: 107103 usec
    Results over last test:
      Probes sent: 5, Probes received: 5, Loss percentage: 0
      Test completed on Sun Sep 13 19:12:34 2015
      Measurement: Round trip time
        Samples: 5, Minimum: 26787 usec, Maximum: 33622 usec, Average: 28176 usec, Peak to peak: 6835 usec, Stddev: 2723 usec, Sum: 140881 usec
    Results over all tests:
      Probes sent: 1448059, Probes received: 1446168, Loss percentage: 0
      Measurement: Round trip time
        Samples: 1446168, Minimum: 26169 usec, Maximum: 278087 usec, Average: 27425 usec, Peak to peak: 251918 usec, Stddev: 3588 usec, Sum: 39660865528 usec
```

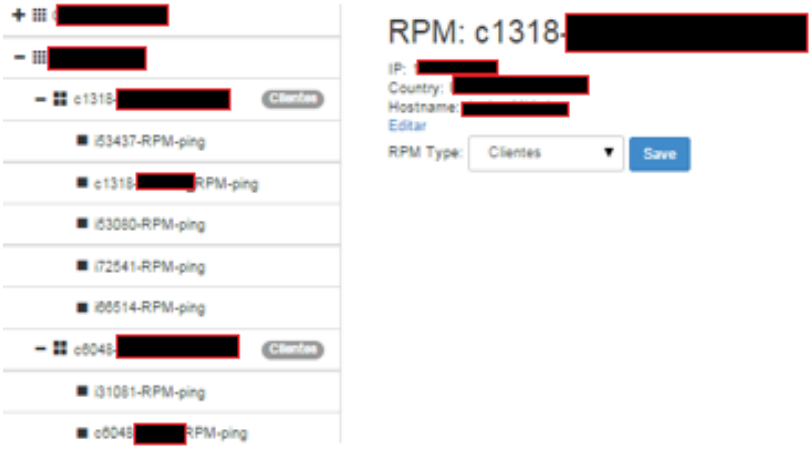
c. Sobre la nomenclatura de las sondas RPM. La configuración de sondas RPM se realizó bajo una nomenclatura específica y estrictamente definida no sólo para identificar fácilmente el monitoreo de servicios (utilizando el nombre de la VRF y números de instalación), sino que para organizar apropiadamente la información en el portal de servicios implementado dentro la empresa de telecomunicaciones. (Ver diseño experimental). A continuación puede verse los resultados del uso de esta nomenclatura en el portal.

- Clasificación de sondas de RPM por suscriptor: La nomenclatura utilizada para el nombre de las sondas, permite organizar los servicios RPM por código de cliente. Es decir, clasifica las sondas por código de suscriptor.

Figura#156: Clasificación de sondas RPM por código de suscriptor

- Pruebas de RPM dentro de sondas RPM: Las pruebas RPM se diferencian de pruebas hacia la central y de pruebas hacia sedes remotas por los caracteres "*c*" e "*i*". La jerarquía misma del equipo permite clasificar las pruebas de RPM dentro de las sondas respectivas.

Figura#157: Pruebas RPM dentro de sondas RPM

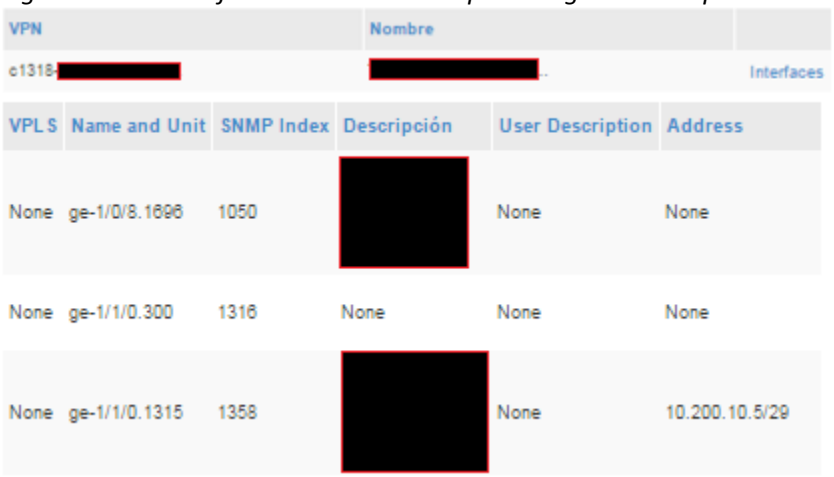


- Clasificación de servicios por VRF centrales: El uso de los "*annotates*" o comentarios sobre la jerarquía de cada prueba RPM dentro de una sonda, permite clasificar todas estas pruebas dentro de las sondas respectivas y en consecuencia la clasificación de todos los servicios que pasan por cada VRF central.

Figura#158: Clasificación de servicios por VRF centrales

- Identificación de centrales por código de suscriptor: El carácter "*c*" dentro de la jerarquía de descripción de subinterfaces hacia las sedes centrales de suscriptores, permite la clasificación de centrales por suscriptor, al ser identificadas por código de cliente.

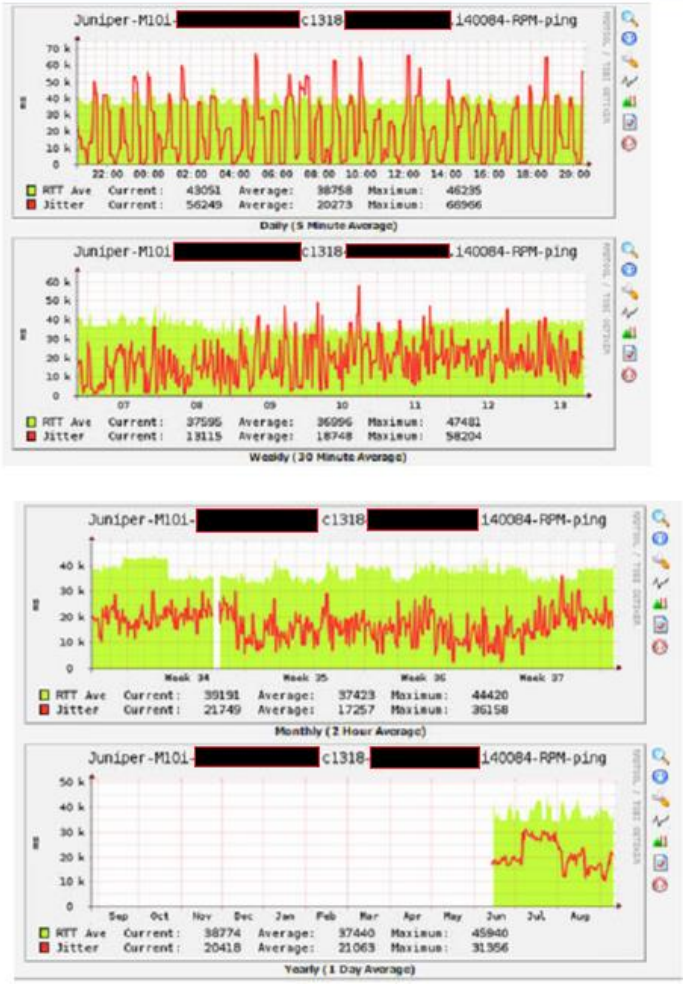
Figura#159: Identificación de centrales por código de suscriptor



3. GENERACIÓN DE GRÁFICAS EN CACTI. Cacti permite el registro actual e histórico de las pruebas de RPM de forma gráfica. Como puede verse en las imágenes, el script utilizado grafica RTT y Jitter. A continuación, se observan las distintas formas de visualizar los gráficos.

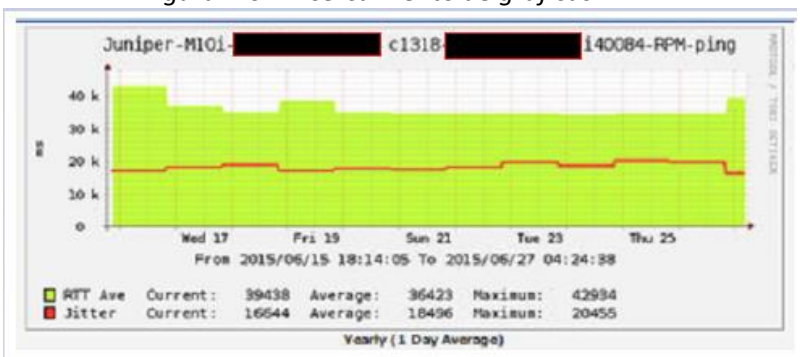
a. Visualización de gráficas temporales e históricas: Cacti muestra gráficas a nivel de temporal (diario, semanal, mensual) e histórico (anual).

Figura#160: Visualización de gráficas RPM temporales e históricas



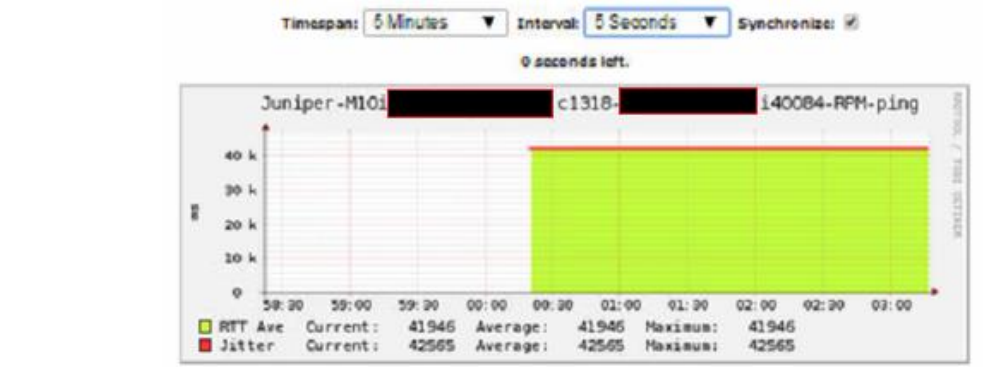
b. Acercamiento de gráficas: Cacti permite un acercamiento a nivel de segundos para cualquier gráfica de RPM temporal o histórica, sin embargo dependiendo de la clasificación temporal de la gráfica debe considerarse la frecuencia del poleo. A continuación puede verse el acercamiento a nivel de días de una gráfica anual (poleo con frecuencia 24 horas).

Figura#161: Acercamiento de gráficas RPM



c. Gráficas en tiempo real: Permite observar el comportamiento del RTT y Jitter de las gráficas RPM a una frecuencia (5 segundos por defecto) y un *timespan* configurables (5 minutos por defecto). Esto puede hacerse desde cualquier tipo de gráfica temporal (pero debe considerarse la frecuencia del poleo).

Figura#162: Gráficas RPM en tiempo real



4. SYSLOG. Los resultados dados por la configuración de archivos syslog para las sondas RPM se dividen en los generados por el umbral de ping perdido y por el umbral de test perdido. Así mismo, se muestran los resultados de envío a un servidor syslog.

a. Logs generados por *PING-PROBE-FAILED*: se configuró un umbral de 1 ping perdido.

Figura#163: Logs por umbral "successive-loss"

```
{master}
jrobles@[redacted]-re0> show log rpm_NOMBRE-SONDA | last 5
Sep 12 23:35:28 [redacted]-re0 rmopd[2033]: PING_PROBE_FAILED: pingCtlOwnerIndex = c1318-NOMBRE-SONDA, pingCtlTestName = i54242-RPM-ping
Sep 13 11:49:52 [redacted]-re0 rmopd[2033]: PING_PROBE_FAILED: pingCtlOwnerIndex = c1318-NOMBRE-SONDA, pingCtlTestName = i54242-RPM-ping
Sep 13 16:12:55 [redacted]-re0 rmopd[2033]: PING_PROBE_FAILED: pingCtlOwnerIndex = c1318-NOMBRE-SONDA, pingCtlTestName = i54242-RPM-ping
Sep 13 18:58:38 [redacted]re0 rmopd[2033]: PING_PROBE_FAILED: pingCtlOwnerIndex = c1318-NOMBRE-SONDA, pingCtlTestName = i55112-RPM-ping
Sep 13 19:17:09 [redacted]re0 rmopd[2033]: PING_PROBE_FAILED: pingCtlOwnerIndex = c1318-NOMBRE-SONDA, pingCtlTestName = i54242-RPM-ping
```

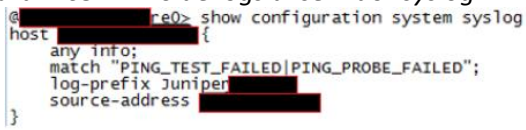
b. Logs generados por *PING-TEST-FAILED:* se configuró un umbral de 15 pruebas de ping perdidas, es decir una prueba RPM completa.

Figura#164: Logs por umbral "total-loss"

```
{master}
jrobles@[redacted]-re0> show log rpm_c2425-NOMBRE-SONDA | last 17
Sep 13 21:47:10 [redacted]-re0 rmopd[2033]: PING_PROBE_FAILED: pingCtlOwnerIndex = c2425-NOMBRE-SONDA, pingCtlTestName = i16840-RPM-ping
Sep 13 21:47:40 [redacted]-re0 rmopd[2033]: PING_PROBE_FAILED: pingCtlOwnerIndex = c2425-NOMBRE-SONDA, pingCtlTestName = i16840-RPM-ping
Sep 13 21:48:10 [redacted]-re0 rmopd[2033]: PING_PROBE_FAILED: pingCtlOwnerIndex = c2425-NOMBRE-SONDA, pingCtlTestName = i16840-RPM-ping
Sep 13 21:48:40 [redacted]-re0 rmopd[2033]: PING_PROBE_FAILED: pingCtlOwnerIndex = c2425-NOMBRE-SONDA, pingCtlTestName = i16840-RPM-ping
Sep 13 21:49:10 [redacted]-re0 rmopd[2033]: PING_PROBE_FAILED: pingCtlOwnerIndex = c2425-NOMBRE-SONDA, pingCtlTestName = i16840-RPM-ping
Sep 13 21:49:40 [redacted]-re0 rmopd[2033]: PING_PROBE_FAILED: pingCtlOwnerIndex = c2425-NOMBRE-SONDA, pingCtlTestName = i16840-RPM-ping
Sep 13 21:50:10 [redacted]-re0 rmopd[2033]: PING_PROBE_FAILED: pingCtlOwnerIndex = c2425-NOMBRE-SONDA, pingCtlTestName = i16840-RPM-ping
Sep 13 21:50:41 [redacted]-re0 rmopd[2033]: PING_PROBE_FAILED: pingCtlOwnerIndex = c2425-NOMBRE-SONDA, pingCtlTestName = i16840-RPM-ping
Sep 13 21:51:11 [redacted]-re0 rmopd[2033]: PING_PROBE_FAILED: pingCtlOwnerIndex = c2425-NOMBRE-SONDA, pingCtlTestName = i16840-RPM-ping
Sep 13 21:51:41 [redacted]-re0 rmopd[2033]: PING_PROBE_FAILED: pingCtlOwnerIndex = c2425-NOMBRE-SONDA, pingCtlTestName = i16840-RPM-ping
Sep 13 21:52:11 [redacted]-re0 rmopd[2033]: PING_PROBE_FAILED: pingCtlOwnerIndex = c2425-NOMBRE-SONDA, pingCtlTestName = i16840-RPM-ping
Sep 13 21:52:41 [redacted]-re0 rmopd[2033]: PING_PROBE_FAILED: pingCtlOwnerIndex = c2425-NOMBRE-SONDA, pingCtlTestName = i16840-RPM-ping
Sep 13 21:53:11 [redacted]-re0 rmopd[2033]: PING_PROBE_FAILED: pingCtlOwnerIndex = c2425-NOMBRE-SONDA, pingCtlTestName = i16840-RPM-ping
Sep 13 21:53:41 [redacted]-re0 rmopd[2033]: PING_PROBE_FAILED: pingCtlOwnerIndex = c2425-NOMBRE-SONDA, pingCtlTestName = i16840-RPM-ping
Sep 13 21:54:12 [redacted]-re0 rmopd[2033]: PING_PROBE_FAILED: pingCtlOwnerIndex = c2425-NOMBRE-SONDA, pingCtlTestName = i16840-RPM-ping
Sep 13 21:54:12 [redacted]-re0 rmopd[2033]: PING_TEST_FAILED: pingCtlOwnerIndex = c2425-NOMBRE-SONDA, pingCtlTestName = i16840-RPM-ping
Sep 13 21:57:12 [redacted]-re0 rmopd[2033]: PING_PROBE_FAILED: pingCtlOwnerIndex = c2425-NOMBRE-SONDA, pingCtlTestName = i16840-RPM-ping
```

c. Servidor Syslog:

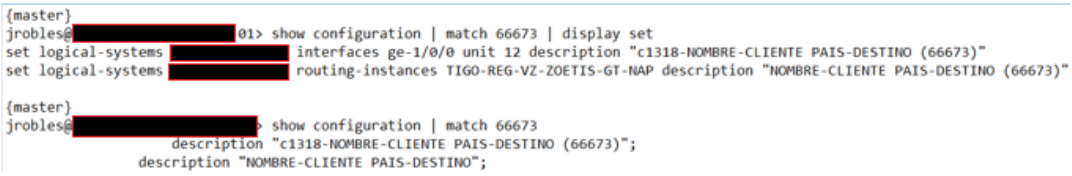
Figura#165: Envío de logs al servidor syslog



5. IDENTIFICACIÓN DE VRFs Y SUBINTERFACES EN PE CENTRAL. Esta identificación se realizó en la jerarquía de descripción tanto para las VRF como para las subinterfaces. Ambas cumplieron el objetivo de identificar todos los servicios que pasan a través de las mismas para llegar a la sede central, y agilizar el proceso de revisión y diagnóstico al operador. Por esta razón, puede observarse que la descripción es básicamente la misma, con la diferencia que las subinterfaces inician con el patrón “cCODIGO_CLIENTE”, para su clasificación en el portal de servicios.

A continuación se muestra cómo la búsqueda por instalación de servicios facilita la ubicación de los mismos en un equipo PE central.

Figura#166: Búsqueda de servicios IP en PE central



B. RESULTADOS DEL SISTEMA DE MONITOREO MPLS VPLS

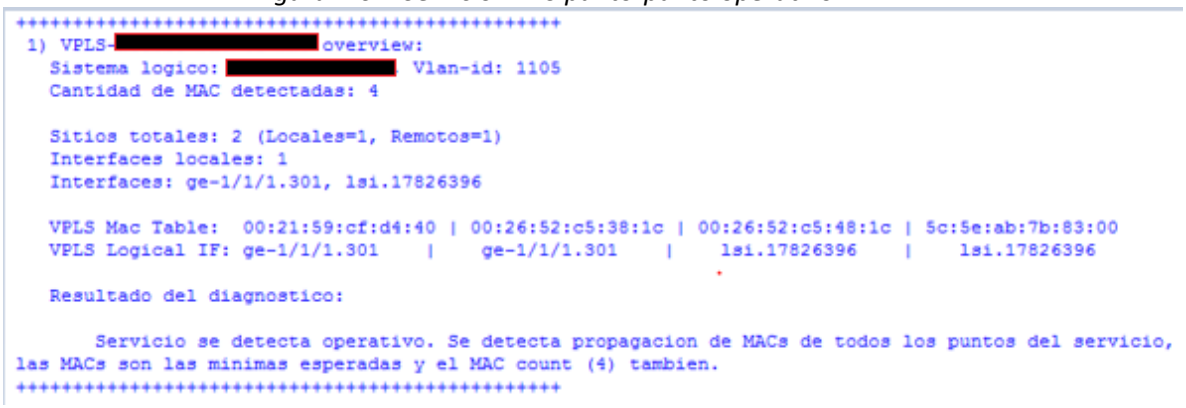
La Figura 99 ilustra una representación gráfica de alto nivel del monitoreo de capa 2. Detalles sobre la misma se discuten en el análisis de resultados.

1. INVESTIGACIÓN DE RECURSOS DE MONITOREO. SNMP es una herramienta ampliamente utilizada para el monitoreo de atributos de equipos de red. Por las razones expuestas en el diseño experimental, se escogió NETCONF como protocolo de comunicación y la explicación se amplía en el análisis de resultados. Adicionalmente, en la sección de anexos, puede observarse cómo instalar las herramientas de monitoreo utilizadas (Python, PyEZ y sus dependencias, habilitación de NETCONF) sobre sistema operativo Windows.

2. PRUEBAS Y ANÁLISIS DEL ALGORITMO. A continuación se detallan los resultados de todos los escenarios posibles que se pueden dar de acuerdo al algoritmo de diagnóstico implementado.

a. Servicio VPLS punto-punto operativo

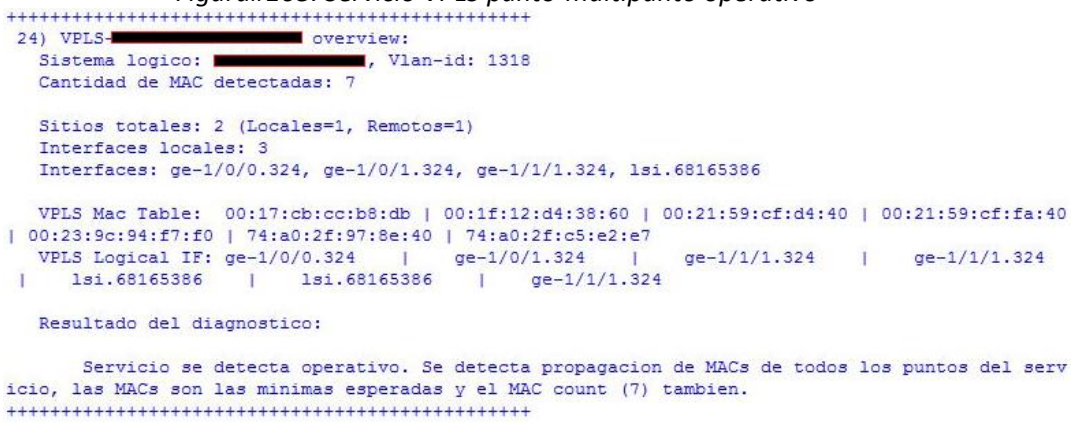
Figura#167: Servicio VPLS punto-punto operativo



```
++++++++++++++++++++++++++++++++++++++++++++++++
 1) VPLS-[redacted] overview:
   Sistema logico: [redacted], Vlan-id: 1105
   Cantidad de MAC detectadas: 4

   Sitios totales: 2 (Locales=1, Remotos=1)
   Interfaces locales: 1
   Interfaces: ge-1/1/1.301, lsi.17826396

   VPLS Mac Table:  00:21:59:cf:d4:40 | 00:26:52:c5:38:1c | 00:26:52:c5:48:1c | 5c:5e:ab:7b:83:00
   VPLS Logical IF: ge-1/1/1.301    |    ge-1/1/1.301    |    lsi.17826396    |    lsi.17826396

   Resultado del diagnostico:

       Servicio se detecta operativo. Se detecta propagacion de MACs de todos los puntos del servicio,
las MACs son las minimas esperadas y el MAC count (4) tambien.
++++++++++++++++++++++++++++++++++++++++++++++++
```

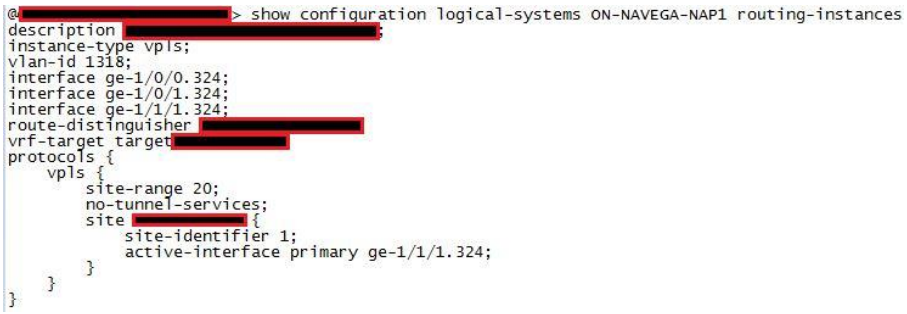
b. Servicio VPLS punto-multipunto operativo.

Figura#168: Servicio VPLS punto-multipunto operativo



```
++++++++++++++++++++++++++++++++++++++++++++++++++
 24) VPLS-[redacted] overview:
   Sistema logico: [redacted], Vlan-id: 1318
   Cantidad de MAC detectadas: 7

   Sitios totales: 2 (Locales=1, Remotos=1)
   Interfaces locales: 3
   Interfaces: ge-1/0/0.324, ge-1/0/1.324, ge-1/1/1.324, lsi.68165386

   VPLS Mac Table:  00:17:cb:cc:b8:db | 00:1f:12:d4:38:60 | 00:21:59:cf:d4:40 | 00:21:59:cf:fa:40
| 00:23:9c:94:f7:f0 | 74:a0:2f:97:8e:40 | 74:a0:2f:c5:e2:e7
   VPLS Logical IF: ge-1/0/0.324    |    ge-1/0/1.324    |    ge-1/1/1.324    |    ge-1/1/1.324
 |    lsi.68165386    |    lsi.68165386    |    ge-1/1/1.324

   Resultado del diagnostico:

       Servicio se detecta operativo. Se detecta propagacion de MACs de todos los puntos del serv
icio, las MACs son las minimas esperadas y el MAC count (7) tambien.
++++++++++++++++++++++++++++++++++++++++++++++++++
```

c. Servicio VPLS con varias interfaces locales y una detectada como activa primaria



```
@[redacted]> show configuration logical-systems ON-NAVEGA-NAP1 routing-instances
description [redacted];
instance-type vpls;
vlan-id 1318;
interface ge-1/0/0.324;
interface ge-1/0/1.324;
interface ge-1/1/1.324;
route-distinguisher [redacted]
vrf-target target[redacted]
protocols {
    vpls {
        site-range 20;
        no-tunnel-services;
        site [redacted] {
            site-identifier 1;
            active-interface primary ge-1/1/1.324;
        }
    }
}
```

```
++++++++++++++++++++++++++++++++++++++++++++++++++
 25) VPLS-[redacted]overview:
   Sistema logico: [redacted], Vlan-id: 1318
   Cantidad de MAC detectadas: 3

   Sitios totales: 2 (Locales=1, Remotos=1)
   Interfaces locales: 1
   Interfaces: ge-1/1/1.324, ge-1/1/1.324, lsi.68165386

   VPLS Mac Table:  00:23:9c:94:f7:f0 | 74:a0:2f:97:8e:40 | 74:a0:2f:c5:e2:e7
   VPLS Logical IF: lsi.68165386    |    lsi.68165386    |    ge-1/1/1.324

   Resultado del diagnostico:

       Servicio se detecta operativo. Se detecta propagacion de MACs de todos
++++++++++++++++++++++++++++++++++++++++++++++++++
```

Figura#168b: Servicio VPLS con varias interfaces locales y una detectada como activa primaria

d. Servicio VPLS con varias interfaces locales y todas activas (no existe activa primaria)

Figura#168c: Servicio VPLS con varias interfaces locales y todas activas (no existe activa primaria)

```
@0[redacted] show configuration logical-systems [redacted] routing-instances
description [redacted];
instance-type vpls;
vlan-id 1083;
interface ge-1/0/0.441;
interface ge-1/0/1.441;
route-distinguisher [redacted]
vrf-target target:[redacted]
protocols {
    vpls {
        site-range 20;
        no-tunnel-services;
        site ce-1 {
            site-identifier 1;
        }
    }
}
++++++++++++++++++++++++++++++++++++++++++++++++++
 18) VPLS[redacted] overview:
   Sistema logico:[redacted], Vlan-id: 1083
   Cantidad de MAC detectadas: 4

   Sitios totales: 2 (Locales=1, Remotos=1)
   Interfaces locales: 2
   Interfaces: ge-1/0/0.441, ge-1/0/1.441, lsi.68165397

   VPLS Mac Table:  00:12:1e:03:a8:20 | 00:17:cb:cc:b8:db | 00:23:33:b8:73:20 | d4:ca:6d:0c:f3:d8
   VPLS Logical IF: ge-1/0/1.441    |    ge-1/0/0.441    |    lsi.68165397    |    lsi.68165397

   Resultado del diagnostico:

       Servicio se detecta operativo. Se detecta propagacion de MACs de todos los puntos del servi
++++++++++++++++++++++++++++++++++++++++++++++++++
```

e. Servicio VPLS no operativo por tener menos direcciones MAC que interfaces lógicas activas.

Figura#169: Servicio VPLS detectado como no operativo (1)

```
++++++++++++++++++++++++++++++++++++++++++++++++++
 37) VPLS[redacted]
   Sistema logico: [redacted] Vlan-id: 1337
   Cantidad de MAC detectadas: 1

   Sitios totales: 3 (Locales=1, Remotos=2)
   Interfaces locales: 1
   Interfaces: ge-1/1/1.1337, lsi.68165489, lsi.68165492

   VPLS Mac Table:  dc:38:e1:a6:57:f0
   VPLS Logical IF: lsi.68165492

   Resultado del diagnostico:

       Este servicio se detecta caido. Existen menos direcciones MAC que sitios/interfaces
asociadas a la VPLS. Debe provenir al menos una MAC por sitio/interfaz.
       No se reciben direcciones MAC de las siguientes interfaces:
         > Interfaz: ge-1/1/1.1337
         > Interfaz: lsi.68165489
++++++++++++++++++++++++++++++++++++++++++++++++++
```

f. Servicio VPLS no operativo por no recibir direcciones MAC de interfaces lógicas remotas (LSI).

Figura#170: Servicio VPLS detectado como no operativo (2)

```
++++++++++++++++++++++++++++++++++++++++++++++++++
 28) VPLS-[redacted] overview:
   Sistema logico: [redacted], Vlan-id: 1356
   Cantidad de MAC detectadas: 5

   Sitios totales: 5 (Locales=1, Remotos=4)
   Interfaces locales: 1
   Interfaces: ge-1/1/0.1356, lsi.17826349, lsi.17826329, lsi.17826330, lsi.17826331

   VPLS Mac Table:  5c:45:27:65:2f:f0 | cc:cc:81:be:b7:5e | cc:cc:81:be:b8:2a | cc:cc:81:be:cb:52 |
fc:48:ef:25:02:3e
   VPLS Logical IF: ge-1/1/0.1356    |    lsi.17826329    |    lsi.17826329    |    lsi.17826331
|    lsi.17826349

   Resultado del diagnostico:

       Este servicio se detecta caido. Existen sitios/interfaces sin direcciones MAC asociadas.
       No se reciben direcciones MAC de las siguientes interfaces:
         > Interfaz: lsi.17826330
++++++++++++++++++++++++++++++++++++++++++++++++++
```

g. Servicio VPLS no operativo por no recibir direcciones MAC de interfaces lógicas locales activas.

Figura#171: Servicio VPLS detectado como no operativo (3)

```
++++++++++++++++++++++++++++++++++++++++++++++++++
 21) VPLS-[redacted] overview:
   Sistema logico: [redacted], Vlan-id: 1074
   Cantidad de MAC detectadas: 2

   Sitios totales: 3 (Locales=1, Remotos=2)
   Interfaces locales: 1
   Interfaces: ge-1/1/0.1074, lsi.17826305, lsi.17826287

   VPLS Mac Table:  00:21:28:e7:ef:cc | 4c:96:14:a2:68:30
   VPLS Logical IF: lsi.17826287    |    lsi.17826305

   Resultado del diagnostico:

       Este servicio se detecta caido. Existen menos direcciones MAC que sitios/interfaces asociadas
a la VPLS. Debe provenir al menos una MAC por sitio/interfaz.
       No se reciben direcciones MAC de las siguientes interfaces:
         > Interfaz: ge-1/1/0.1074
++++++++++++++++++++++++++++++++++++++++++++++++++
```

h. Servicio VPLS operativo pero con menos direcciones MAC de las esperadas lo que implica cambios en direcciones MAC originales.

Figura#172: Servicio VPLS operativo condicional (1)

```
++++++++++++++++++++++++++++++++++++++++++++++++++
 14) VPLS-[redacted] overview:
   Sistema logico: [redacted], Vlan-id: 1339
   Cantidad de MAC detectadas: 131

   Sitios totales: 2 (Locales=1, Remotos=1)
   Interfaces locales: 1
   Interfaces: ge-1/1/1.805, lsi.17826399

   Resultado del diagnostico:

       Servicio probablemente operativo. Se detecta MACs de ambos puntos pero se tienen las siguientes observaciones:
         > Existen menos direcciones MAC de las esperadas, el numero de MACs default es: 132
         > La direccion MAC: 00:1b:54:41:8b:12, deberia provenir de la interfaz logica: lsi.17826399
         > La direccion MAC: 00:50:56:8a:47:0b, deberia provenir de la interfaz logica: lsi.17826399
         > La direccion MAC: 00:50:56:90:53:fe, deberia provenir de la interfaz logica: lsi.17826399
         > La direccion MAC: 00:50:56:91:01:d6, deberia provenir de la interfaz logica: lsi.17826399
         > La direccion MAC: 00:50:56:91:46:f8, deberia provenir de la interfaz logica: lsi.17826399
         > La direccion MAC: 00:50:56:97:09:b9, deberia provenir de la interfaz logica: lsi.17826399
         > La direccion MAC: 00:50:56:97:25:e6, deberia provenir de la interfaz logica: lsi.17826399
         > La direccion MAC: 18:a9:05:75:f0:e6, deberia provenir de la interfaz logica: lsi.17826399
++++++++++++++++++++++++++++++++++++++++++++++++++
```

i. Servicio VPLS operativo pero con más direcciones MAC de las esperadas y con cambios en direcciones MAC originales.

Figura#173: Servicio VPLS operativo condicional (2)

```
++++++++++++++++++++++++++++++++++++++++++++++++
 3) VPLS-[redacted] overview:
   Sistema logico: [redacted] Vlan-id: 1357
   Cantidad de MAC detectadas: 77

   Sitios totales: 2 (Locales=1, Remotos=1)
   Interfaces locales: 1
   Interfaces: ge-1/1/1.1357, lsi.17826357
   Resultado del diagnostico:

       Servicio probablemente operativo. Se detecta MACs de ambos puntos pero se tienen las siguientes observaciones:
         > Existen mas direcciones MAC de las esperadas, el numero de MACs default es: 71
         > La direccion MAC: 00:1b:4f:13:1c:86, deberia provenir de la interfaz logica: lsi.17826357
         > La direccion MAC: 00:1b:4f:13:1c:d0, deberia provenir de la interfaz logica: lsi.17826357
         > La direccion MAC: 00:1b:4f:13:1c:db, deberia provenir de la interfaz logica: lsi.17826357
         > La direccion MAC: 00:1b:4f:13:1c:f0, deberia provenir de la interfaz logica: lsi.17826357
         > La direccion MAC: 00:1b:4f:13:1c:f4, deberia provenir de la interfaz logica: lsi.17826357
         > La direccion MAC: 00:1b:4f:13:1d:45, deberia provenir de la interfaz logica: lsi.17826357
         > La direccion MAC: 00:1b:4f:13:28:6c, deberia provenir de la interfaz logica: lsi.17826357
         > La direccion MAC: 00:1b:4f:13:28:ea, deberia provenir de la interfaz logica: lsi.17826357
         > La direccion MAC: 00:1b:4f:30:03:c6, deberia provenir de la interfaz logica: lsi.17826357
         > La direccion MAC: 00:1b:4f:30:03:e5, deberia provenir de la interfaz logica: lsi.17826357
         > La direccion MAC: 70:38:ee:d4:3b:72, deberia provenir de la interfaz logica: lsi.17826357
         > La direccion MAC: 9c:8e:99:39:e5:bf, deberia provenir de la interfaz logica: lsi.17826357
++++++++++++++++++++++++++++++++++++++++++++++++
```

j. Servicio VPLS operativo pero con más direcciones MAC de las esperadas y sin cambios en direcciones MAC originales (actualización de registros).

Figura#174: Servicio VPLS operativo condicional (3)

```
++++++++++++++++++++++++++++++++++++++++++++++++
 21) VPL[redacted]verview:
   Sistema logico:[redacted]lan-id: 1186
   Cantidad de MAC detectadas: 30

   Sitios totales: 2 (Locales=1, Remotos=1)
   Interfaces locales: 1
   Interfaces: ge-1/1/0.1186, lsi.17826265

   VPLS Mac Table:  00:05:dc:88:f8:1a | 00:0d:66:34:f8:1b | 00:0e:5e:32:75:3e | 00:0f:34:8e:25:14 | 00:11:5d:f9:1e:0e | 00
:12:43:79:78:1b | 00:16:c8:9c:ff:00 | 00:19:e2:dc:a1:fe | 00:21:1b:19:24:dc | 00:24:38:9b:92:93 | 00:25:84:7c:73:80 | 00:8
c:fa:12:fa:d2 | 00:d0:63:2a:b4:19 | 00:e0:52:c2:ea:dd | 40:55:39:4f:64:cf | 4c:5e:0c:23:cf:07 | 4c:5e:0c:23:d5:3c | 4c:5e:
0c:e6:f7:e0 | 4c:5e:0c:e9:49:56 | 58:8d:09:66:10:1b | 74:46:a0:27:0a:c0 | 74:46:a0:27:ae:e0 | 84:2b:2b:48:09:ef | 84:2b:2b
:48:0d:93 | 84:2b:2b:48:11:88 | 84:2b:2b:48:1f:f8 | 88:f0:77:4b:81:10 | b0:fa:eb:6d:4d:80 | b0:fa:eb:6d:4d:81 | b0:fa:eb:6
d:4d:83
   VPLS Logical IF: lsi.17826265    |    lsi.17826265    |    lsi.17826265    |    lsi.17826265    |    lsi.17826265    |
  lsi.17826265    |    lsi.17826265    |    lsi.17826265    |    lsi.17826265    |    ge-1/1/0.1186    |    lsi.17826265
 |    lsi.17826265    |    lsi.17826265    |    lsi.17826265    |    lsi.17826265    |    lsi.17826265    |    lsi.1782626
5    |    lsi.17826265    |    lsi.17826265    |    lsi.17826265    |    lsi.17826265    |    lsi.17826265    |    lsi.178
26265    |    lsi.17826265    |    lsi.17826265    |    lsi.17826265    |    lsi.17826265    |    lsi.17826265    |    lsi
.17826265    |    lsi.17826265

   Resultado del diagnostico:

       Servicio probablemente operativo. Se detecta MACs de ambos puntos pero se tienen las siguientes observaciones:
         > Existen mas direcciones MAC de las esperadas, el numero de MACs default es: 28
         > Sin embargo, las direcciones MAC originales se mantienen por lo que se detecta servicio operativo y se procede
a actualizar la informacion de esta VPLS.
++++++++++++++++++++++++++++++++++++++++++++++++
```

k. Servicio VPLS con misma cantidad de direcciones MAC pero con cambios en direcciones MAC originales.

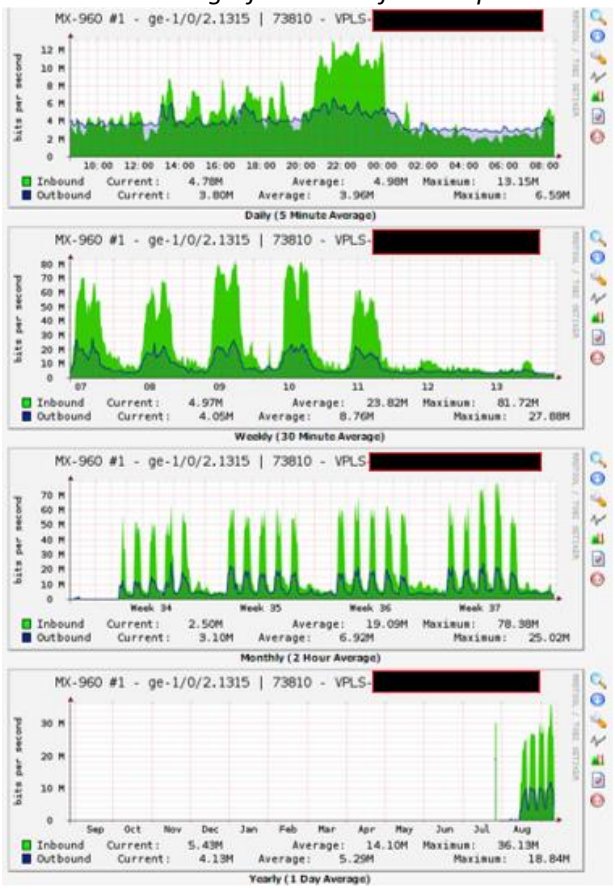
Figura#175: Servicio VPLS operativo condicional (4)

```
++++++++++++++++++++++++++++++++++++++++++++++
1) VPLS-[redacted] overview:
   Sistema logico: [redacted]  Vlan-id: 1339
   Cantidad de MAC detectadas: 135

   Sitios totales: 2 (Locales=1, Remotos=1)
   Interfaces locales: 1
   Interfaces: ge-1/1/1.805, lsi.17826399

   Resultado del diagnostico:

      Servicio probablemente operativo. Se detecta MACs de ambos puntos pero se tienen las siguientes
observaciones:
        > El numero de MACs es el esperado: 135, pero se detectan cambios de MACS.
        > La direccion MAC: 00:50:56:6f:ed:5b, deberia provenir de la interfaz logica: lsi.17826399
        > La direccion MAC: 00:50:56:7f:fc:1c, deberia provenir de la interfaz logica: lsi.17826399
        > La direccion MAC: 00:50:56:90:00:00, deberia provenir de la interfaz logica: lsi.17826399
        > La direccion MAC: 00:a0:c9:00:01:01, deberia provenir de la interfaz logica: lsi.17826399
++++++++++++++++++++++++++++++++++++++++++++++++
```

3. GRÁFICAS DE TRÁFICO. Al igual que para servicio RPM, Cacti permite el registro actual e histórico de gráficas de tráfico, a través de un poleo vía SNMP. Como puede verse en las imágenes a continuación, lo que se grafica es tráfico de salida y de entrada de las interfaces lógicas implicadas.

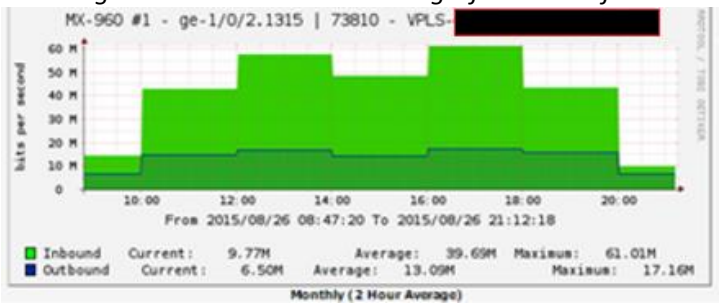
a. Visualización de gráficas temporales e históricas: Cacti muestra gráficas a nivel de temporal (diario, semanal, mensual) e histórico (anual).

Figura#176: Visualización de gráficas de tráfico temporales e históricas



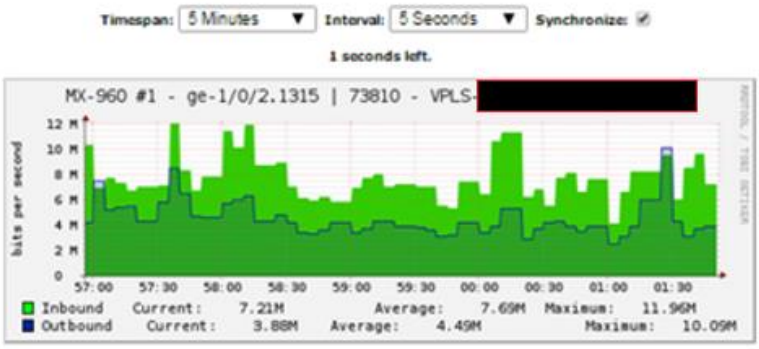
b. Acercamiento de gráficas: A continuación puede verse el acercamiento a nivel de horas de una gráfica mensual (poleo con frecuencia 2 horas).

Figura#177: Acercamiento de gráficas de tráfico



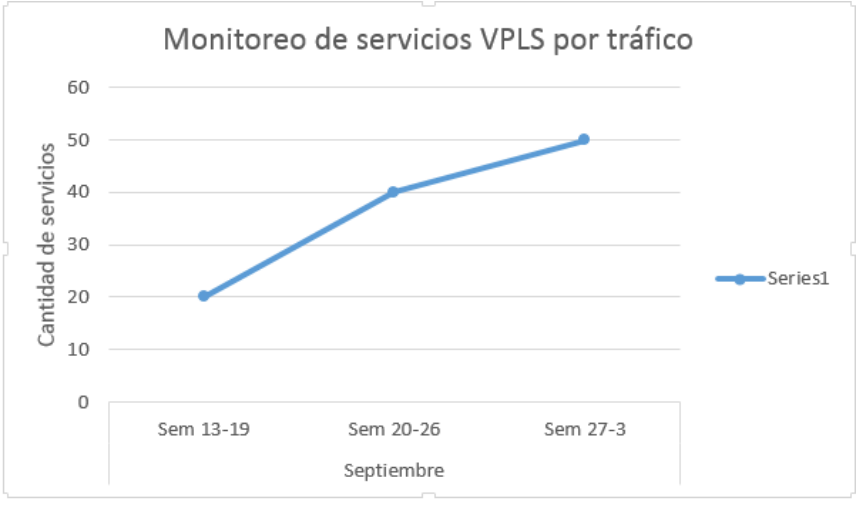
c. Gráficas en tiempo real: Al igual que para gráficas de RPM, esto puede hacerse desde cualquier tipo de gráfica temporal (pero debe considerarse la frecuencia del poleo).

Figura#178: Gráficas de tráfico en tiempo real



Para cada servicio VPLS se graficó la interfaz asociada del PE remoto y la interfaz respectiva del PE central. A continuación se muestra el avance en el monitoreo de servicios VPLS por generación de gráficas de tráfico en función del tiempo.

Figura#179: Avance en monitoreo de servicios VPLS por gráficas de tráfico



4. IDENTIFICACIÓN DE VPLS Y SUBINTERFACES CENTRALES. Esta identificación se realizó en la jerarquía de descripción tanto para las VPLS como para las subinterfaces centrales asociadas, y cumplió el mismo objetivo que la identificación de servicios IP.

A continuación se muestra cómo una búsqueda de un servicio VPLS.

Figura#180: Búsqueda de servicios VPLS en PE central

```
{master}
jrobles@[redacted]MX960-RE0-01> show configuration logical-systems [redacted] | match 73810 | display set
set logical-systems [redacted] interfaces ge-1/0/2 unit 1315 description "c43365-NOMBRE_CLIENTE PAIS DESTINO (73810)"
set logical-systems [redacted] routing-instances NOMBRE-VPLS description "NOMBRE_CLIENTE PAIS DESTINO (73810)"

{master}
jrobles@[redacted]MX960-RE0-01> show configuration logical-systems [redacted] | match 73810
            description "c43365-NOMBRE_CLIENTE PAIS DESTINO (73810)";

        description "NOMBRE_CLIENTE PAIS DESTINO (73810)";
```

5. RESULTADOS DE RPM SORE SERVICIOS VPLS. Una vez se le configura RPM a un servicio VPLS, los resultados son los mismos que los observados para un servicio IP.

a. Resultados RPM históricos

Figura#181: Resultados RPM históricos de servicio VPLS

```
jrobles@jun.m10i.nap.mia.dat.02-re0> show services rpm history-results owner c43365-NOMBRE-VPLS
    Owner, Test                 Probe received              Round trip time
    c43365-NOMBRE-VPLS, i73810-RPM-ping Mon Sep 14 17:55:51 2015            37748 usec
    c43365-NOMBRE-VPLS, i73810-RPM-ping Mon Sep 14 17:56:21 2015            37753 usec
    c43365-NOMBRE-VPLS, i73810-RPM-ping Mon Sep 14 17:56:52 2015            37704 usec
    c43365-NOMBRE-VPLS, i73810-RPM-ping Mon Sep 14 17:57:22 2015            37723 usec
    c43365-NOMBRE-VPLS, i73810-RPM-ping Mon Sep 14 17:57:52 2015            37741 usec
    c43365-NOMBRE-VPLS, i73810-RPM-ping Mon Sep 14 17:58:22 2015            37763 usec
    c43365-NOMBRE-VPLS, i73810-RPM-ping Mon Sep 14 17:58:52 2015            37694 usec
    c43365-NOMBRE-VPLS, i73810-RPM-ping Mon Sep 14 17:59:22 2015            37810 usec
```

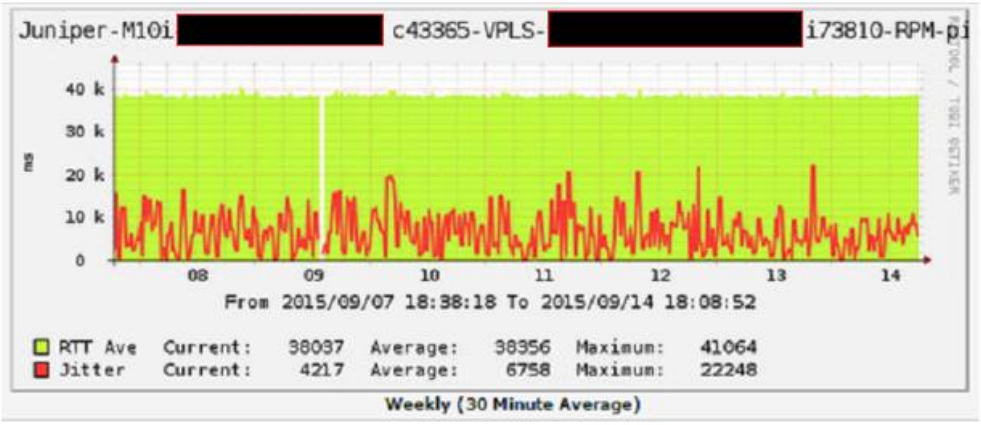
b. Resultados RPM de sonda

Figura#182: Resultados RPM de sonda de servicio VPLS

```
jrobles@jun.m10i.nap.mia.dat.02-re0> show services rpm probe-results owner c43365-NOMBRE-VPLS test i73810-RPM-ping
    Owner: c43365-NOMBRE-VPLS, Test: i73810-RPM-ping
    Target address: 10.200.10.1, Probe type: icmp-ping
    Routing Instance Name: NOMBRE-VPLS
    Test size: 15 probes
    Probe results:
      Response received, Mon Sep 14 18:27:29 2015, No hardware timestamps
      Rtt: 37749 usec
    Results over current test:
      Probes sent: 12, Probes received: 12, Loss percentage: 0
      Measurement: Round trip time
        Samples: 12, Minimum: 37690 usec, Maximum: 59377 usec, Average: 39549 usec, Peak to peak: 21687 usec, Stddev: 5979 usec, Sum: 474585 usec
    Results over last test:
      Probes sent: 15, Probes received: 15, Loss percentage: 0
      Test completed on Mon Sep 14 18:19:27 2015
      Measurement: Round trip time
        Samples: 15, Minimum: 37684 usec, Maximum: 37786 usec, Average: 37723 usec, Peak to peak: 102 usec, Stddev: 29 usec, Sum: 565849 usec
    Results over all tests:
      Probes sent: 29802, Probes received: 29798, Loss percentage: 0
      Measurement: Round trip time
        Samples: 29798, Minimum: 37627 usec, Maximum: 78524 usec, Average: 38366 usec, Peak to peak: 40897 usec, Stddev: 2942 usec, Sum: 1143226087 usec
```

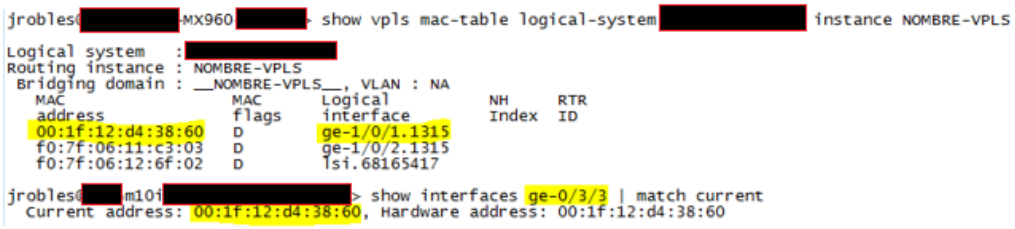
c. Gráficas RPM

Figura#183: Gráficas RPM de servicio VPLS



d. Reconocimiento de direcciones MAC del equipo de monitoreo: Puede observarse cómo, al establecer una conexión IP y transferir información (RPM-ping), se reconoce la dirección MAC de la interfaz del equipo de monitoreo donde se levanta dicha conexión, en la tabla VPLS respectiva.

Figura#184: Reconocimiento de direcciones MAC al levantar conectividad IP



6. RESULTADOS DE PLAN DE PRUEBAS DEL SISTEMA DE MONITOREO VPLS. A continuación, los resultados globales y semanales de las pruebas en escenarios reales del sistema de monitoreo de VPLS. Adicionalmente, se incluye un ejemplo de la detección de caída de un servicio VPLS por generación de alarma roja.

Cuadro#1: Resultados globales de pruebas de sistema de monitoreo VPLS

Fecha	Caidos	Detectados con anticipación	Detectados sin anticipación	No detectados	Efectividad (%)	Trafico máximo	Trafico	Tiempo de detección
Semana 1	17	16	0	1	94.12	30bps in, 40bps out	30bps in, 0bps out	7 min
Semana 2	9	8	0	1	88.89	60bps in, 40bps out	30bps in, 7.5bps out	2 min
Semana 3	9	8	1	0	88.89	2000bps in, 30bps out	30bps in, 0bps out	2 min
Total	**35**	**32**	**1**	**2**	**91.43**	**2000bps in, 40bps out**	**30bps in, 0bps out**	**3.6 min**

Cuadro#2 Resultados globales de alarmas generadas por el sistema de monitoreo VPLS

Alarmas	Cantidad	De caída	En Falso	%Efectividad
Rojas	29	29	0	100.00
Naranjas	145	4	141	2.76
Amarillas	14	0	14	0.00
Verdes	76	0	76	0.00

Cuadro#3: Resultados semana 1 de pruebas de sistema de monitoreo VPLS

Fecha	Caidos	Detectados con anticipación	Detectados sin anticipación	No detectados	Efectividad (%)	Trafico máximo	Trafico	Tiempo de detección
26-oct	3	3	0	0	100	30bps in, 40bps out	30 in, 25 out	7 min
28-oct	7	7	0	0	100	30 bps in, 30bps out	30bps in, 0 out	10 min
29-oct	1	1	0	0	100	30bps in, 0 out	30bps in, 0 out	0 min
30-oct	2	2	0	0	100	30bps in, 40bps out	30bps in, 20bps out	10 min
31-oct	4	3	0	1	75	30bps in, 40bps out	30bps in, 0bps out	7 min
Total	**17**	**16**	**0**	**1**	**94.12**	**30bps in, 40bps out**	**30bps in, 0bps out**	**7 min**

Cuadro#4 Resultados semana 1 de alarmas generadas por el sistema de monitoreo VPLS

Alarmas	Cantidad	De caída	En Falso	%Efectividad
Rojas	13	13	0	100.00
Naranjas	51	3	48	5.88
Amarillas	5	0	5	0.00
Verdes	31	0	31	0.00

Cuadro#5: Resultados semana 2 de pruebas de sistema de monitoreo VPLS

Fecha	Caidos	Detectados con anticipación	Detectados sin anticipación	No detectados	Efectividad (%)	Trafico máximo	Trafico	Tiempo de detección
02-nov	2	1	0	1	50	30bps in, 40bps out	30bps in, 20bps out	0 min
03-nov	2	2	0	0	100	60 bps in, 0bps out	45bps in, 0 out	1.5min
04-nov	1	1	0	0	100	30bps in, 0 out	30bps in, 0 out	0 min
05-nov	2	2	0	0	100	30bps in, 0bps out	30bps in, 0bps out	7.5min
06-nov	2	2	0	0	100	30bps in, 30bps out	15bps in, 15bps out	0 min
Total	**9**	**8**	**0**	**1**	**88.89**	**60bps in, 40bps out**	**30bps in, 7.5bps out**	**2 min**

Cuadro#6 Resultados semana 2 de alarmas generadas por el sistema de monitoreo VPLS

Alarmas	Cantidad	De caída	En Falso	%Efectividad
Rojas	8	8	0	100
Naranjas	45	0	45	0
Amarillas	4	0	4	0
Verdes	22	0	22	0

Cuadro#7: Resultados semana 3 de pruebas de sistema de monitoreo VPLS

Fecha	Caidos	Detectados con anticipación	Detectados sin anticipación	No detectados	Efectividad (%)	Trafico máximo	Trafico	Tiempo de detección
09-nov	3	2	1	0	66.67	2000bps in, 0bps out	2000bps in, 0bps out	5 min
10-nov	1	1	0	0	100	30bps in, 0 out	30bps in, 0 out	0 min
11-nov	3	3	0	0	100	30bps in, 30bps out	20bps in, 20bps out	2 min
12-nov	1	1	0	0	100	30bps in, 0bps out	30bps in, 0bps out	0 min
13-nov	1	1	0	0	100	30bps in, 0bps out	30bps in, 0bps out	0 min
Total	**9**	**8**	**1**	**0**	**88.89**	**2000bps in, 30bps out**	**30bps in, 0bps out**	**2 min**

Cuadro#8 Resultados semana 3 de alarmas generadas por el sistema de monitoreo VPLS

Alarmas	Cantidad	De caída	En Falso	%Efectividad
Rojas	8	8	0	100.00
Naranjas	49	1	48	2.04
Amarillas	5	0	5	0.00
Verdes	23	0	23	0.00

Cuadro#9 Servicios VPLS activos y depurados

Equipo	Servicios VPLS	Activos	Depurados
NAP1	52	48	4
NAP2	44	36	8
ER	35	34	1
Total	**131**	**118**	**13**

El siguiente es un ejemplo de un servicio VPLS detectado fuera de operación durante la etapa de pruebas.

28 de octubre, horario de pruebas: 14:00-18:00hrs GMT-6

Tipo de Alarma: Roja
Escenario 2: Servicio VPLS no operativo por no recibir direcciones MAC de sitio remoto.

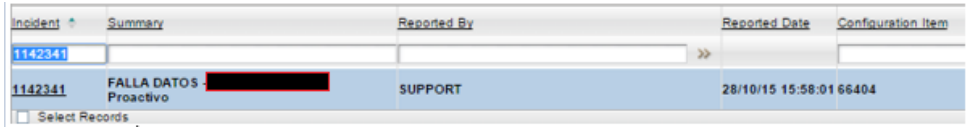
Suscriptor con servicio VPLS dado por el ISP
Internacional: USA-Honduras
ISP ID: 66404
Ticket ISP: 1142341; Ticket Suscriptor: No reportó.

Evento: Se generó el ticket 1142341 proactivo. Operación local en Honduras confirmó falla general por un corte de fibra en su red Metro que afectó varios servicios. El tráfico de los servicios se conmutó manualmente a una ruta de protección para solventar el evento.

- *Horario de caída: 15:40 GMT-6*
- *Horario de detección ISP: 15:55GMT-6*
- *Horario reporte Suscriptor: N/A*
- *Horario de resolución: 16:30GMT-6.*

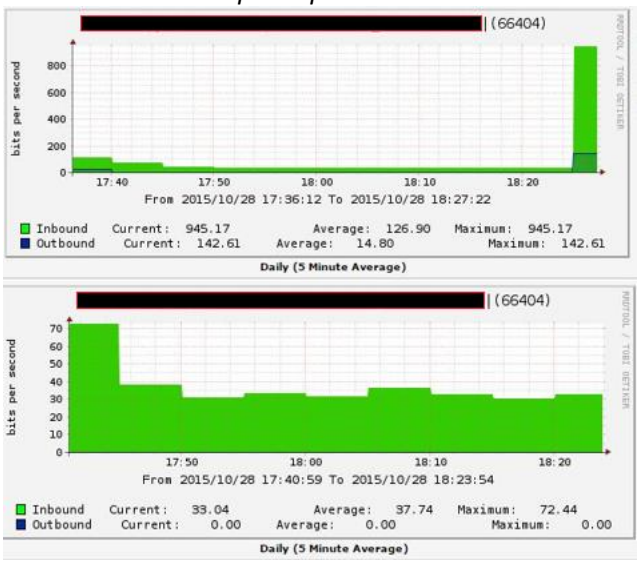
Ticket (GMT-6):

Figura#185: Ticket generado durante etapa de pruebas de sistema de monitoreo capa 2



Tráfico (GMT-4)

Figura#186: Baja de tráfico vista durante etapa de pruebas de sistema de monitoreo capa 2



7. CONFIGURACIÓN DE VPLS DE PRUEBAS

Figura#187: Anuncio y recepción de la comunidad VPLS a través de iBGP en NAP1

```
{master}
jrobles@[redacted]NAP-01[redacted]> show route logical-system [redacted] advertising-protocol bgp [redacted].134.2 | match 8060
  [redacted]134.1:8060:1:1/96
  [redacted]134.1:8060:1:1/96

{master}
jrobles@[redacted]NAP-01[redacted]> show route logical-system [redacted] receive-protocol bgp [redacted]134.2 | match 8060
  [redacted]134.2:8060:5:1/96
  [redacted]134.2:8060:5:1/96
```

Figura#188: Anuncio y recepción de la comunidad VPLS a través de iBGP en NAP2

```
{master}
jrobles@[redacted]NAP-02[redacted]> show route logical-system [redacted] advertising-protocol bgp [redacted]134.1 | match 8060
  [redacted]134.2:8060:5:1/96
  [redacted]134.2:8060:5:1/96

{master}
jrobles@[redacted]NAP-02[redacted]> show route logical-system [redacted] receive-protocol bgp [redacted]134.1 | match 8060
  [redacted]134.1:8060:1:1/96
  [redacted]134.1:8060:1:1/96
```

Figura#189: Establecimiento de la VPLS en NAP1

```
{master}
jrobles@[redacted]NAP-01[redacted]> show vpls connections extensive logical-system [redacted] instance VPLS-MONITOREO-PRUEBA_TESIS

Instance: VPLS-MONITOREO-PRUEBA_TESIS
  Local site: Prueba-1 (1)
    Number of local interfaces: 3
    Number of local interfaces up: 3
    IRB interface present: no
    ge-1/0/0.804
    ge-1/0/1.804
    ge-1/1/1.804
    lsi.68165605        5          Intf - vpls VPLS-MONITOREO-PRUEBA_TESIS local site 1 remote site 5
    Label-base        Offset     Size  Range     Preference
    327769            1          8     8         100
    connection-site           Type  St     Time last up          # Up trans
    5                         rmt   Up     Nov  4 15:56:36 2015           1
      Remote PE: [redacted]134.2, Negotiated control-word: No
      Incoming label: 327773, Outgoing label: 327825
      Local interface: lsi.68165605, Status: Up, Encapsulation: VPLS
        Description: Intf - vpls VPLS-MONITOREO-PRUEBA_TESIS local site 1 remote site 5
    Connection History:
        Nov  4 15:56:36 2015  status update timer
        Nov  4 15:56:36 2015  loc intf up                lsi.68165605
        Nov  4 15:56:36 2015  PE route changed
        Nov  4 15:56:36 2015  Out lbl Update                   327825
        Nov  4 15:56:36 2015  In lbl Update                    327773
        Nov  4 15:56:36 2015  loc intf down
```

Figura#190: Establecimiento de VPLS en NAP2

```
{master}
jrobles@[redacted]NAP-02[redacted]> show vpls connections extensive logical-system [redacted] instance VPLS-MONITOREO-PRUEBA_TESIS

Instance: VPLS-MONITOREO-PRUEBA_TESIS
  Local site: Prueba-2 (5)
    Number of local interfaces: 2
    Number of local interfaces up: 2
    IRB interface present: no
    ge-1/0/1.804
    ge-1/1/1.804
    lsi.17826496        1          Intf - vpls VPLS-MONITOREO-PRUEBA_TESIS local site 5 remote site 1
    Label-base        Offset     Size  Range     Preference
    327825            1          8     8         100
    connection-site           Type  St     Time last up          # Up trans
    1                         rmt   Up     Nov  4 15:56:36 2015           1
      Remote PE: [redacted]134.1, Negotiated control-word: No
      Incoming label: 327825, Outgoing label: 327773
      Local interface: lsi.17826496, Status: Up, Encapsulation: VPLS
        Description: Intf - vpls VPLS-MONITOREO-PRUEBA_TESIS local site 5 remote site 1
    Connection History:
        Nov  4 15:56:36 2015  status update timer
        Nov  4 15:56:36 2015  loc intf up                lsi.17826496
        Nov  4 15:56:36 2015  PE route changed
        Nov  4 15:56:36 2015  Out lbl Update                   327773
        Nov  4 15:56:36 2015  In lbl Update                    327825
        Nov  4 15:56:36 2015  loc intf down
```

Figura#191: Tabla MAC de VPLS en NAP1

```
{master}
jrobles@[redacted]NAP-01[redacted]> show vpls mac-table logical-system [redacted] instance VPLS-MONITOREO-PRUEBA_TESIS

Logical system    : [redacted]
Routing instance : VPLS-MONITOREO-PRUEBA_TESIS
 Bridging domain : __VPLS-MONITOREO-PRUEBA_TESIS__, VLAN : 804
   MAC                 MAC      Logical          NH     RTR
   address             flags    interface        Index  ID
   00:17:cb:cc:b8:46   D        ge-1/1/1.804
   00:17:cb:cc:b8:db   D        ge-1/0/0.804
   00:1f:12:d4:38:60   D        ge-1/0/1.804
   00:1f:12:d4:38:9e   D        lsi.68165605
   00:1f:12:d4:38:de   D        lsi.68165605
```

Figura#192: Tabla MAC de VPLS en NAP2

```
{master}
jrobles@[redacted]NAP-02[redacted]> show vpls mac-table logical-system [redacted] instance VPLS-MONITOREO-PRUEBA_TESIS

Logical system    : [redacted]
Routing instance : VPLS-MONITOREO-PRUEBA_TESIS
 Bridging domain : __VPLS-MONITOREO-PRUEBA_TESIS__, VLAN : 804
   MAC                 MAC      Logical          NH     RTR
   address             flags    interface        Index  ID
   00:17:cb:cc:b8:46   D        lsi.17826496
   00:17:cb:cc:b8:db   D        lsi.17826496
   00:1f:12:d4:38:60   D        lsi.17826496
   00:1f:12:d4:38:9e   D        ge-1/1/1.804
   00:1f:12:d4:38:de   D        ge-1/0/1.804
```

Figura#193: Direcciones MAC de los equipos de monitoreo M10I#1, M10I#4

```
{master}
jrobles@[redacted]> show interfaces fe-0/2/8 | match hardware
  Current address: 00:17:cb:cc:b8:46, Hardware address: 00:17:cb:cc:b8:46

{master}
jrobles@[redacted]> show interfaces ge-1/3/0 | match hardware
  Current address: 00:17:cb:cc:b8:db, Hardware address: 00:17:cb:cc:b8:db

jrobles@[redacted]> show interfaces fe-1/1/1 | match hardware
  Current address: 00:1f:12:d4:38:9e, Hardware address: 00:1f:12:d4:38:9e

jrobles@[redacted]> show interfaces ge-0/3/3 | match hardware
  Current address: 00:1f:12:d4:38:60, Hardware address: 00:1f:12:d4:38:60

jrobles@[redacted]> show interfaces ge-1/3/3 | match hardware
  Current address: 00:1f:12:d4:38:de, Hardware address: 00:1f:12:d4:38:de
```

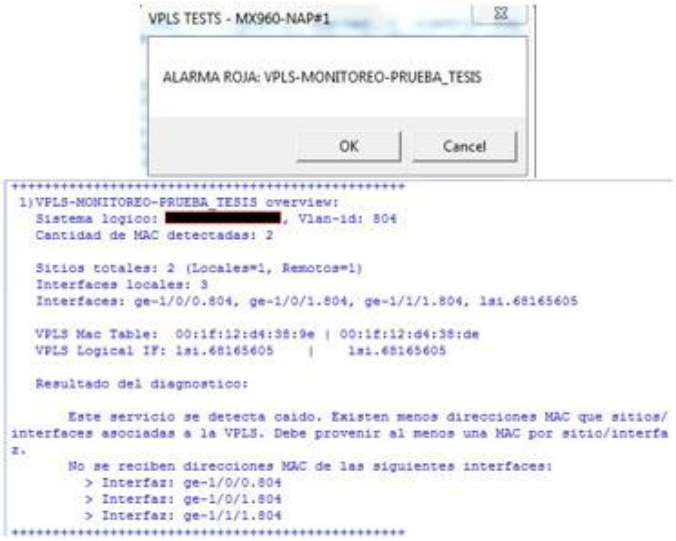
Figura#194: RPM en equipos de monitoreo M10I#1, M10I#4 para VPLS de pruebas

```
{master}
jrobles@[redacted]I-re0> show services rpm history-results owner VPLS-MONITOREO-PRUEBA_TESIS_RPM test VPLS-MONITOREO_RPM-ping | last 10
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping Sat Nov  7 23:56:12 2015            1256 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping Sat Nov  7 23:56:42 2015            1294 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping Sat Nov  7 23:57:13 2015            1313 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping Sat Nov  7 23:57:43 2015            1223 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping Sun Nov  8 00:00:13 2015           39004 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping Sun Nov  8 00:00:43 2015            1288 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping Sun Nov  8 00:01:13 2015            1238 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping Sun Nov  8 00:01:43 2015            1199 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping Sun Nov  8 00:02:14 2015            1212 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping Sun Nov  8 00:02:44 2015            1133 usec

{master}
jrobles@[redacted]I-re0> show services rpm history-results owner VPLS-MONITOREO-PRUEBA_TESIS_RPM test VPLS-MONITOREO_RPM-ping2 | last 10
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping2 Sat Nov  7 23:56:12 2015            1153 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping2 Sat Nov  7 23:56:42 2015            1279 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping2 Sat Nov  7 23:57:13 2015            1171 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping2 Sat Nov  7 23:57:43 2015            1175 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping2 Sun Nov  8 00:00:13 2015           38951 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping2 Sun Nov  8 00:00:43 2015            1195 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping2 Sun Nov  8 00:01:13 2015            1137 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping2 Sun Nov  8 00:01:43 2015            1189 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping2 Sun Nov  8 00:02:14 2015            1215 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping2 Sun Nov  8 00:02:44 2015            1091 usec
jrobles@[redacted]dat.02-re0> show services rpm history-results owner VPLS-MONITOREO-PRUEBA_TESIS_RPM test VPLS-MONITOREO_RPM-ping | last 10
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping Sat Nov  7 23:51:00 2015            1078 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping Sat Nov  7 23:51:31 2015            1208 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping Sat Nov  7 23:52:01 2015            1132 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping Sat Nov  7 23:52:31 2015            1059 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping Sat Nov  7 23:53:01 2015            1097 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping Sat Nov  7 23:53:31 2015            1127 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping Sat Nov  7 23:54:01 2015            1194 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping Sat Nov  7 23:54:32 2015            1180 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping Sat Nov  7 23:57:02 2015            1059 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping Sat Nov  7 23:57:32 2015            1179 usec

jrobles@[redacted]dat.02-re0> show services rpm history-results owner VPLS-MONITOREO-PRUEBA_TESIS_RPM test VPLS-MONITOREO_RPM-ping2 | last 10
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping2 Sat Nov  7 23:51:31 2015            1049 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping2 Sat Nov  7 23:52:01 2015             947 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping2 Sat Nov  7 23:52:31 2015             997 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping2 Sat Nov  7 23:53:01 2015            1034 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping2 Sat Nov  7 23:53:31 2015            1063 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping2 Sat Nov  7 23:54:01 2015            1132 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping2 Sat Nov  7 23:54:32 2015            1114 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping2 Sat Nov  7 23:57:02 2015            1008 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping2 Sat Nov  7 23:57:32 2015            1021 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping2 Sat Nov  7 23:58:02 2015             936 usec

jrobles@[redacted]dat.02-re0> show services rpm history-results owner VPLS-MONITOREO-PRUEBA_TESIS_RPM test VPLS-MONITOREO_RPM-ping3 | last 10
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping3 Sat Nov  7 23:51:31 2015             708 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping3 Sat Nov  7 23:52:01 2015             715 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping3 Sat Nov  7 23:52:31 2015             693 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping3 Sat Nov  7 23:53:01 2015             621 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping3 Sat Nov  7 23:53:31 2015             692 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping3 Sat Nov  7 23:54:01 2015             730 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping3 Sat Nov  7 23:54:32 2015             647 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping3 Sat Nov  7 23:57:02 2015             631 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping3 Sat Nov  7 23:57:32 2015             669 usec
    VPLS-MONITOREO-PRUEBA_TESIS_RPM, VPLS-MONITOREO_RPM-ping3 Sat Nov  7 23:58:02 2015             635 usec
```

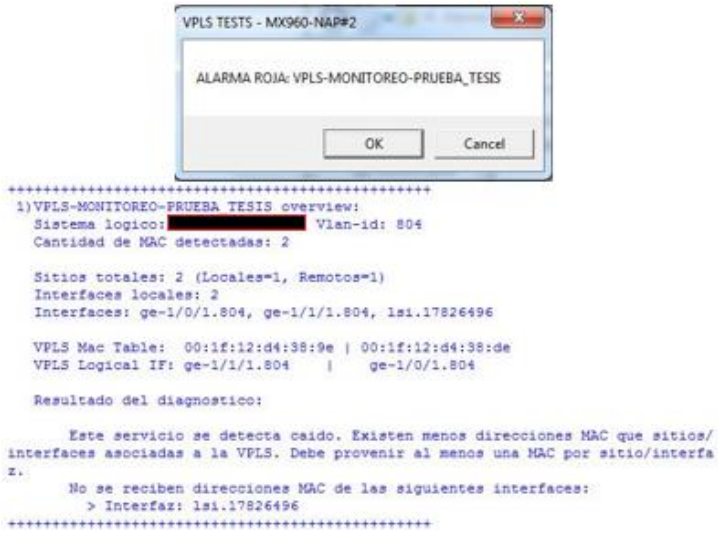
8. PRUEBAS TÉCNICAS SOBRE VPLS CONFIGURADA

a. Escenario 1: Servicio normalizado (con 5 “sedes remotas”) previo al inicio. Durante la prueba, deshabilitar todas las subinterfaces de M10Is con IPs: 10.10.10.1/29, 10.10.10.3/29 y 10.10.10.4/29.

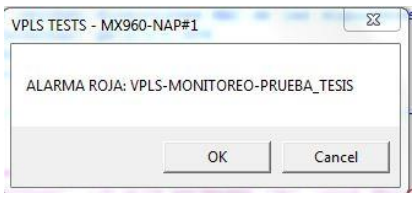
Figura#195: Pruebas técnicas sobre VPLS configurada: Escenario 1, NAP1



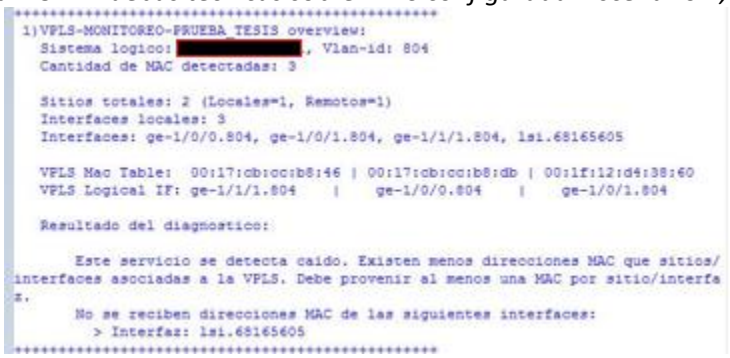
Figura#196: Pruebas técnicas sobre VPLS configurada: Escenario 1, NAP2



b. Escenario 2: Servicio normalizado (con 5 “sedes remotas”) previo al inicio. Durante la prueba, deshabilitar subinterfaces M10Is 10.10.10.2/29 y 10.10.10.5/29.



Figura#197: Pruebas técnicas sobre VPLS configurada: Escenario 2, NAP1

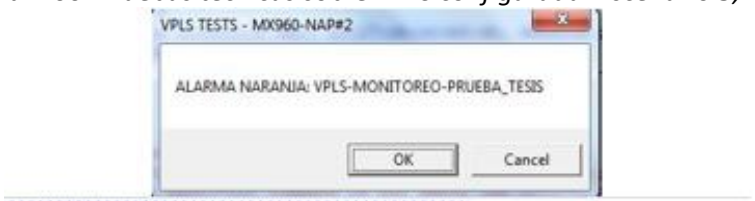


Figura#198: Pruebas técnicas sobre VPLS configurada: Escenario 2, NAP2

c. Escenario 3: Servicio normalizado (con 5 “sedes remotas”) previo al inicio. Durante la prueba, deshabilitar una única subinterfaz de M10Is entre 10.10.10.1/29, .3 ó .4.

Figura#199: Pruebas técnicas sobre VPLS configurada: Escenario 3, NAP1

Figura#200: Pruebas técnicas sobre VPLS configurada: Escenario 3, NAP2




```
++++++++++++++++++++++++++++++++++++++++++++++++
 1)VPLS-MONITOREO-PRUEBA_TESIS overview:
   Sistema logico: [redacted], Vlan-id: 804
   Cantidad de MAC detectadas: 4

   Sitios totales: 2 (Locales=1, Remotos=1)
   Interfaces locales: 2
   Interfaces: ge-1/0/1.804, ge-1/1/1.804, lsi.17826496

   VPLS Mac Table:  00:17:cb:cc:b8:46 | 00:1f:12:d4:38:60 | 00:1f:12:d4:38:9e
| 00:1f:12:d4:38:de
   VPLS Logical IF: lsi.17826496    |    lsi.17826496    |    ge-1/1/1.804
|    ge-1/0/1.804

   Resultado del diagnostico:

       Servicio probablemente operativo. Se detecta MACs de ambos puntos pero
se tienen las siguientes observaciones:
         > Existen menos direcciones MAC de las esperadas, el numero de MACs
default es: 5
         > La direccion MAC: 00:17:cb:cc:b8:db, deberia provenir de la interf
az logica: lsi.17826496
--------------------------------------------------------------------------------
```

d. Escenario 4: Servicio normalizado (con 5 “sedes remotas”) previo al inicio. Durante la prueba, deshabilitar una única subinterfaz M10Is entre 10.10.10.2/29 ó 10.10.10.5/29.

Figura#201: Pruebas técnicas sobre VPLS configurada: Escenario 4, NAP1


```
++++++++++++++++++++++++++++++++++++++++++++++++
 1)VPLS-MONITOREO-PRUEBA_TESIS overview:
   Sistema logico: [redacted], Vlan-id: 804
   Cantidad de MAC detectadas: 4

   Sitios totales: 2 (Locales=1, Remotos=1)
   Interfaces locales: 3
   Interfaces: ge-1/0/0.804, ge-1/0/1.804, ge-1/1/1.804, lsi.68165605

   VPLS Mac Table:  00:17:cb:cc:b8:46 | 00:17:cb:cc:b8:db | 00:1f:12:d4:38:60
| 00:1f:12:d4:38:de
   VPLS Logical IF: ge-1/1/1.804    |    ge-1/0/0.804    |    ge-1/0/1.804
|    lsi.68165605

   Resultado del diagnostico:

       Servicio probablemente operativo. Se detecta MACs de ambos puntos pero
se tienen las siguientes observaciones:
         > Existen menos direcciones MAC de las esperadas, el numero de MACs
default es: 5
         > La direccion MAC: 00:1f:12:d4:38:9e, deberia provenir de la interf
az logica: lsi.68165605
--------------------------------------------------------------------------------
```

Figura#202: Pruebas técnicas sobre VPLS configurada: Escenario 4, NAP2

e. Escenario 5: Servicio con 4 “sedes remotas” deshabilitando una subinterfaz de M10is entre 10.10.10.1/29, 10.10.10.3/29 ó 10.10.10.4/29 previo al inicio. Durante la prueba, habilitar la subinterfaz deshabilitada y deshabilitar alguna de ellas que estuviera habilitadas al inicio. Por ejemplo: Deshabilitar 10.10.10.1/29 previo al inicio, y durante la prueba habilitarla pero al mismo tiempo deshabilitar la 10.10.10.3/29.

Figura#203: Pruebas técnicas sobre VPLS configurada: Escenario 5, NAP1 previo

Figura#204: Pruebas técnicas sobre VPLS configurada: Escenario 5, NAP1 posterior

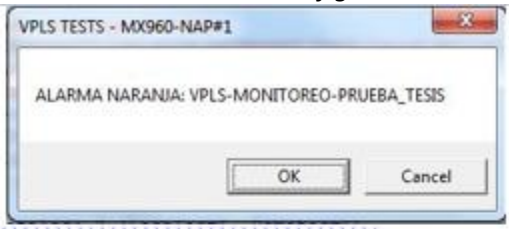
Figura#205: Pruebas técnicas sobre VPLS configurada: Escenario 5, NAP2

f. Escenario 6: Servicio con 4 “sedes remotas” deshabilitando una subinterfaz de M10Is entre 10.10.10.2/29 ó 10.10.10.5/29 previo al inicio. Durante la prueba, habilitar la subinterfaz deshabilitada y deshabilitar alguna de ellas que estuviera habilitadas al inicio.

Figura#206: Pruebas técnicas sobre VPLS configurada: Escenario 6, NAP2 previo

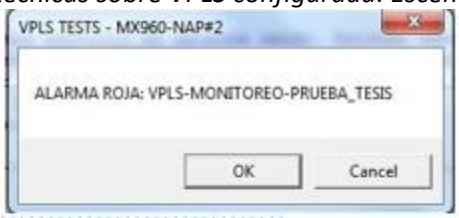

```
+++++++++++++++++++++++++++++++++++++++++++++++++
 1)VPLS-MONITOREO-PRUEBA_TESIS overview:
   Sistema logico: [redacted], Vlan-id: 804
   Cantidad de MAC detectadas: 4

   Sitios totales: 2 (Locales=1, Remotos=1)
   Interfaces locales: 2
   Interfaces: ge-1/0/1.804, ge-1/1/1.804, lsi.17826496

   VPLS Mac Table:  00:17:cb:cc:b8:46 | 00:17:cb:cc:b8:db | 00:1f:12:d4:38:60 | 00:1
f:12:d4:38:de
   VPLS Logical IF: lsi.17826496    |    lsi.17826496    |    lsi.17826496    |    g
e-1/0/1.804

   Resultado del diagnostico:

      Este servicio se detecta caido. Existen menos direcciones MAC que sitios/inte
rfaces asociadas a la VPLS. Debe provenir al menos una MAC por sitio/interfaz.
       No se reciben direcciones MAC de las siguientes interfaces:
         > Interfaz: ge-1/1/1.804
+++++++++++++++++++++++++++++++++++++++++++++++++
```

Figura#207: Pruebas técnicas sobre VPLS configurada: Escenario 6, NAP2 previo




```
+++++++++++++++++++++++++++++++++++++++++++++++++
 1)VPLS-MONITOREO-PRUEBA_TESIS overview:
   Sistema logico: [redacted], Vlan-id: 804
   Cantidad de MAC detectadas: 4

   Sitios totales: 2 (Locales=1, Remotos=1)
   Interfaces locales: 3
   Interfaces: ge-1/0/0.804, ge-1/0/1.804, ge-1/1/1.804, lsi.68165605

   VPLS Mac Table:  00:17:cb:cc:b8:46 | 00:17:cb:cc:b8:db | 00:1f:12:d4:3
8:60 | 00:1f:12:d4:38:9e
   VPLS Logical IF: ge-1/1/1.804    |    ge-1/0/0.804    |    ge-1/0/1.80
4    |    lsi.68165605

   Resultado del diagnostico:

      Servicio probablemente operativo. Se detecta MACs de ambos puntos
pero se tienen las siguientes observaciones:
        > El numero de MACs es el esperado: 4, pero se detectan cambios
de MACS.
        > La direccion MAC: 00:1f:12:d4:38:de, deberia provenir de la in
terfaz logica: lsi.68165605
-------------------------------------------------------------------------
```

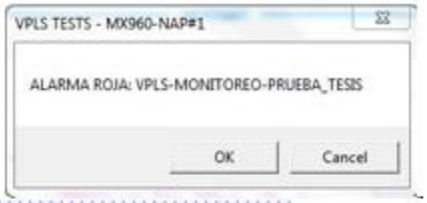
Figura#208: Pruebas técnicas sobre VPLS configurada: Escenario 6, NAP2 posterior




```
++++++++++++++++++++++++++++++++++++++++++++++++
 1)VPLS-MONITOREO-PRUEBA_TESIS overview:
   Sistema logico: [redacted] Vlan-id: 804
   Cantidad de MAC detectadas: 4

   Sitios totales: 2 (Locales=1, Remotos=1)
   Interfaces locales: 2
   Interfaces: ge-1/0/1.804, ge-1/1/1.804, lsi.17826496

   VPLS Mac Table:  00:17:cb:cc:b8:46 | 00:17:cb:cc:b8:db | 00:1f:12:d4:38:60 | 00:1
f:12:d4:38:9e
   VPLS Logical IF: lsi.17826496    |    lsi.17826496    |    lsi.17826496    |    g
e-1/1/1.804

   Resultado del diagnostico:

       Este servicio se detecta caido. Existen menos direcciones MAC que sitios/inte
rfaces asociadas a la VPLS. Debe provenir al menos una MAC por sitio/interfaz.
       No se reciben direcciones MAC de las siguientes interfaces:
         > Interfaz: ge-1/0/1.804
++++++++++++++++++++++++++++++++++++++++++++++++
```

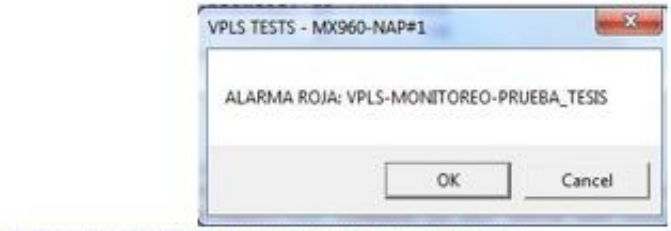
g. Escenario 7: Servicio con 3 “sedes remotas” deshabilitando dos subinterfaces de M10is entre 10.10.10.1/29, 10.10.10.3/29 ó 10.10.10.4/29 previo al inicio. Durante la prueba, habilitar las dos subinterfaces deshabilitadas y deshabilitar la que estuviera habilitada al inicio. Por ejemplo: Deshabilitar 10.10.10.1/29 y 10.10.10.3/29 previo al inicio, y durante la prueba habilitarlas pero al mismo tiempo deshabilitar la 10.10.10.4/29.

Figura#209: Pruebas técnicas sobre VPLS configurada: Escenario 7, NAP1 previo




```
++++++++++++++++++++++++++++++++++++++++++++++++
 1)VPLS-MONITOREO-PRUEBA_TESIS overview:
   Sistema logico: [redacted] Vlan-id: 804
   Cantidad de MAC detectadas: 3

   Sitios totales: 2 (Locales=1, Remotos=1)
   Interfaces locales: 3
   Interfaces: ge-1/0/0.804, ge-1/0/1.804, ge-1/1/1.804, lsi.68165605

   VPLS Mac Table:  00:1f:12:d4:38:60 | 00:1f:12:d4:38:9e | 00:1f:12:d4:38:de
   VPLS Logical IF: ge-1/0/1.804    |    lsi.68165605    |    lsi.68165605

   Resultado del diagnostico:

       Este servicio se detecta caido. Existen menos direcciones MAC que sitios/
interfaces asociadas a la VPLS. Debe provenir al menos una MAC por sitio/interfa
z.
       No se reciben direcciones MAC de las siguientes interfaces:
         > Interfaz: ge-1/0/0.804
         > Interfaz: ge-1/1/1.804
++++++++++++++++++++++++++++++++++++++++++++++++
```

Figura#210: Pruebas técnicas sobre VPLS configurada: Escenario 7, NAP1 posterior



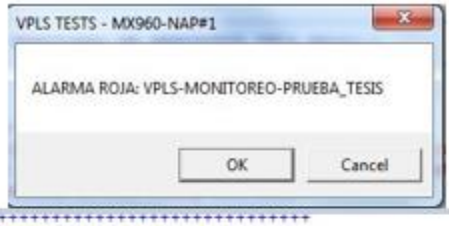

```
++++++++++++++++++++++++++++++++++++++++++++++++
 1)VPLS-MONITOREO-PRUEBA_TESIS overview:
   Sistema logico: [redacted], Vlan-id: 804
   Cantidad de MAC detectadas: 4

   Sitios totales: 2 (Locales=1, Remotos=1)
   Interfaces locales: 3
   Interfaces: ge-1/0/0.804, ge-1/0/1.804, ge-1/1/1.804, lsi.68165605

   VPLS Mac Table:  00:17:cb:cc:b8:46 | 00:17:cb:cc:b8:db | 00:1f:12:d4:38:9e |
00:1f:12:d4:38:de
   VPLS Logical IF: ge-1/1/1.804    |    ge-1/0/0.804    |    lsi.68165605    |
   lsi.68165605

   Resultado del diagnostico:

       Este servicio se detecta caido. Existen menos direcciones MAC que sitios/
interfaces asociadas a la VPLS. Debe provenir al menos una MAC por sitio/interfa
z.
       No se reciben direcciones MAC de las siguientes interfaces:
         > Interfaz: ge-1/0/1.804
++++++++++++++++++++++++++++++++++++++++++++++++
```

Figura#211: Pruebas técnicas sobre VPLS configurada: Escenario 7, NAP2


```
++++++++++++++++++++++++++++++++++++++++++++++++
 1)VPLS-MONITOREO-PRUEBA_TESIS overview:
   Sistema logico: [redacted], Vlan-id: 804
   Cantidad de MAC detectadas: 4

   Sitios totales: 2 (Locales=1, Remotos=1)
   Interfaces locales: 2
   Interfaces: ge-1/0/1.804, ge-1/1/1.804, lsi.17826496

   VPLS Mac Table:  00:17:cb:cc:b8:46 | 00:17:cb:cc:b8:db | 00:1f:12:d4:38:9e |
00:1f:12:d4:38:de
   VPLS Logical IF: lsi.17826496    |    lsi.17826496    |    ge-1/1/1.804    |
   ge-1/0/1.804

   Resultado del diagnostico:

       Servicio probablemente operativo. Se detecta MACs de ambos puntos pero se
 tienen las siguientes observaciones:
        > Existen mas direcciones MAC de las esperadas, el numero de MACs defau
lt es: 3
        > La direccion MAC: 00:1f:12:d4:38:60, deberia provenir de la interfaz
logica: lsi.17826496
--------------------------------------------------------------------------------
```

h. Escenario 8: Servicio con 4 “sedes remotas” deshabilitando una subinterfaz de M10is entre 10.10.10.1/29, 10.10.10.3/29 ó 10.10.10.4/29 previo al inicio. Durante la prueba, únicamente habilitarla. Por ejemplo: Deshabilitar 10.10.10.1/29 previo al inicio, y durante la prueba habilitarla.

Figura#212: Pruebas técnicas sobre VPLS configurada: Escenario 8, NAP1 previo



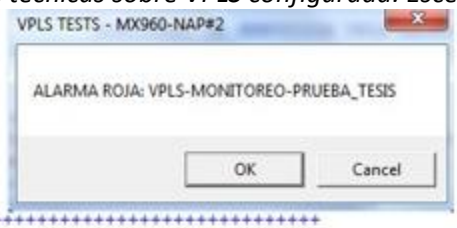

```
++++++++++++++++++++++++++++++++++++++++++++++++
 1)VPLS-MONITOREO-PRUEBA_TESIS overview:
   Sistema logico: [redacted], Vlan-id: 804
   Cantidad de MAC detectadas: 4

   Sitios totales: 2 (Locales=1, Remotos=1)
   Interfaces locales: 3
   Interfaces: ge-1/0/0.804, ge-1/0/1.804, ge-1/1/1.804, lsi.68165605

   VPLS Mac Table:  00:17:cb:cc:b8:db | 00:1f:12:d4:38:60 | 00:1f:12:d4:38:9e |
00:1f:12:d4:38:de
   VPLS Logical IF: ge-1/0/0.804    |    ge-1/0/1.804    |    lsi.68165605    |
   lsi.68165605

   Resultado del diagnostico:

       Este servicio se detecta caido. Existen menos direcciones MAC que sitios/
interfaces asociadas a la VPLS. Debe provenir al menos una MAC por sitio/interfa
z.
       No se reciben direcciones MAC de las siguientes interfaces:
         > Interfaz: ge-1/1/1.804
++++++++++++++++++++++++++++++++++++++++++++++++
```

Figura#213: Pruebas técnicas sobre VPLS configurada: Escenario 8, NAP2


```
++++++++++++++++++++++++++++++++++++++++++++++++
 1)VPLS-MONITOREO-PRUEBA_TESIS overview:
   Sistema logico: [redacted], Vlan-id: 804
   Cantidad de MAC detectadas: 5

   Sitios totales: 2 (Locales=1, Remotos=1)
   Interfaces locales: 2
   Interfaces: ge-1/0/1.804, ge-1/1/1.804, lsi.17826496

   VPLS Mac Table:  00:17:cb:cc:b8:46 | 00:17:cb:cc:b8:db | 00:1f:12:d4:38:60 |
00:1f:12:d4:38:9e | 00:1f:12:d4:38:de
   VPLS Logical IF: lsi.17826496    |    lsi.17826496    |    lsi.17826496    |
   ge-1/1/1.804    |    ge-1/0/1.804

   Resultado del diagnostico:

       Servicio probablemente operativo. Se detecta MACs de ambos puntos pero se
 tienen las siguientes observaciones:
         > Existen mas direcciones MAC de las esperadas, el numero de MACs defau
lt es: 4
         > Sin embargo, las direcciones MAC originales se mantienen por lo que s
e detecta servicio operativo y se procede a actualizar la informacion de esta VP
LS.
++++++++++++++++++++++++++++++++++++++++++++++++
```

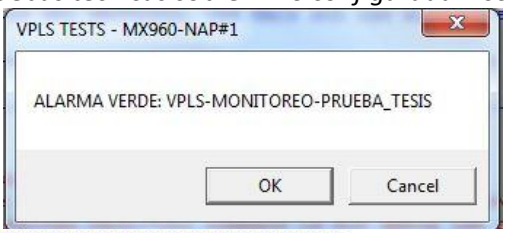
i. Escenario 9: Servicio con 4 “sedes remotas” deshabilitando una subinterfaz de M10is entre 10.10.10.2/29 ó 10.10.10.5/29 previo al inicio. Durante la prueba, únicamente habilitarla. Por ejemplo: Deshabilitar 10.10.10.2/29 previo al inicio, y durante la prueba habilitarla.

Figura#214: Pruebas técnicas sobre VPLS configurada: Escenario 9, NAP2 previo




```
++++++++++++++++++++++++++++++++++++++++++++++++
 1)VPLS-MONITOREO-PRUEBA_TESIS overview:
   Sistema logico: [redacted] Vlan-id: 804
   Cantidad de MAC detectadas: 4

   Sitios totales: 2 (Locales=1, Remotos=1)
   Interfaces locales: 2
   Interfaces: ge-1/0/1.804, ge-1/1/1.804, lsi.17826496

   VPLS Mac Table:  00:17:cb:cc:b8:46 | 00:17:cb:cc:b8:db | 00:1f:12:d4:38:60 |
00:1f:12:d4:38:de
   VPLS Logical IF: lsi.17826496    |    lsi.17826496    |    lsi.17826496    |
   ge-1/0/1.804

   Resultado del diagnostico:

       Este servicio se detecta caido. Existen menos direcciones MAC que sitios/
interfaces asociadas a la VPLS. Debe provenir al menos una MAC por sitio/interfa
z.
       No se reciben direcciones MAC de las siguientes interfaces:
         > Interfaz: ge-1/1/1.804
++++++++++++++++++++++++++++++++++++++++++++++++
```

Figura#215: Pruebas técnicas sobre VPLS configurada: Escenario 9, NAP1




```
++++++++++++++++++++++++++++++++++++++++++++++++
 1)VPLS-MONITOREO-PRUEBA_TESIS overview:
   Sistema logico: ON-NAVEGA-NAP1, Vlan-id: 804
   Cantidad de MAC detectadas: 5

   Sitios totales: 2 (Locales=1, Remotos=1)
   Interfaces locales: 3
   Interfaces: ge-1/0/0.804, ge-1/0/1.804, ge-1/1/1.804, lsi.68165605

   VPLS Mac Table:  00:17:cb:cc:b8:46 | 00:17:cb:cc:b8:db | 00:1f:12:d4:38:60 |
00:1f:12:d4:38:9e | 00:1f:12:d4:38:de
   VPLS Logical IF: ge-1/1/1.804    |    ge-1/0/0.804    |    ge-1/0/1.804    |
   lsi.68165605    |    lsi.68165605

   Resultado del diagnostico:

       Servicio probablemente operativo. Se detecta MACs de ambos puntos pero se
 tienen las siguientes observaciones:
        > Existen mas direcciones MAC de las esperadas, el numero de MACs defau
lt es: 4
        > Sin embargo, las direcciones MAC originales se mantienen por lo que s
e detecta servicio operativo y se procede a actualizar la informacion de esta VP
LS.
++++++++++++++++++++++++++++++++++++++++++++++++
```

9. PRUEBAS DE TIEMPO DE VIDA DE DIRECCIONES MAC. Se muestra el tiempo (en segundos) de vida de direcciones MAC, tanto para ser perdidas de las tablas MAC VPLS como para ser recuperadas, a nivel de sitio local y remoto.

Cuadro#10 Tiempos de vida de direcciones MAC de VPLS en escenarios prácticos

MAC AGING TIME TESTS					
VPLS	**Tipo de prueba**	**Local pérdida**	**Remoto pérdida**	**Loc. Recuperación**	**R. Recuperación**
Servicio VPLS1	Escenario 1	10	553	5	6
Servicio VPLS1	Escenario 2	7	590	4	5
Servicio VPLS2	Escenario 1	6	290	4	6
Servicio VPLS2	Escenario 2	6	275	4	4
VPLS-PRUEBA	Escenario 1.a	240	300	3	4
VPLS-PRUEBA	Escenario 1.b	3	3	27	27
VPLS-PRUEBA	Escenario 2.a	290	530	2	3
VPLS-PRUEBA	Escenario 2.b	3	3	26	26
VPLS-PRUEBA	Escenario 3	300	300	3	3

VIII. ANÁLISIS DE RESULTADOS (DISCUSIÓN)

El objetivo del trabajo, desarrollar e implementar un sistema de monitoreo que permita detectar incidentes sobre servicios que se prestan por medio de enlaces de datos privados Ethernet e IP dentro de una red MPLS, mejorando el análisis y disminuyendo el tiempo de detección y diagnóstico de fallas, con el fin de comprobar una mejora en la calidad de soporte e imagen de la empresa, estuvo dividido en una serie de objetivos específicos previamente detallados en este trabajo, para su debido cumplimiento, y cuyos resultados se analizan y discuten a continuación, llegando a conclusiones concisas.

La naturaleza de los servicios MPLS IP-VPN contrasta con la de los servicios MPLS-VPLS, tal y como se ha enfatizado a lo largo de este trabajo escrito. Esto obliga a buscar distintos subsistemas de monitoreo para los servicios de cada uno de estos dos tipos. Aunque, como se pudo ver en la sección de diseño experimental y de resultados, es posible aplicar el mismo sistema usado en servicios IP a servicios VPLS, existen razones que llevan al uso del sistema determinado y finalmente utilizado para estos enlaces de capa 2. La diferencia entre servicios Ethernet e IP, radica en que un servicio MPLS IPVN está conformado por distintos segmentos de red IP en sus distintos tramos, por lo que el aprovechar la conectiviad vía ping es esencial. En cambio, un servicio MPLS VPLS al emular una conexión Ethernet, le garantiza al suscriptor transparencia en su servicio, siendo el proveedor un medio únicamente de transporte que no utiliza conectividad IP ni protocolos de ruteo. Se mencionó anteriormente, como una analogía, que la red del proveedor funciona virtualmente como un switch para este tipo de servicios, en donde los clientes utilizan sus CE como hosts que se conectan a los distintos puertos del switch y establecen comunicación a nivel IP. El switch únicamente se vale de su tabla MAC para determinar la conectividad en sus puertos. Esto obligó a buscar una forma de aprovechar el reconocimiento de direcciones MAC para establecer el monitoreo de VPLS. Existieron complementos importantes para cada tipo de sistema. En el caso de IP, fue el uso de gráficas, generación de logs, de una nomeclatura estricta y de la identificación de los servicios. En el caso Ethernet, la generación de gráficas de tráfico (que debe ser considerada más que un complemento), identificación de servicios e incluso, una forma alterna de monitoreo. A continuación, se discute a detalle sobre ambos sistemas, sus complementos y las debidas comparaciones entre ambos.

El monitoreo de enlaces MPLS IP-VPN tuvo como eje principal la aplicación del servicio RPM, el cual se configuró desde un equipo de monitoreo hacia uno de los PE del servicio, específicamente el PE central. La Figura 67 muestra un diagrama de alto nivel sobre este monitoreo. El equipo de monitoreo puede verse fuera de la red MPLS del proveedor, ya que el objetivo fue aplicar el RPM lejos de los equipos de transmisión del servicio, dejando que estos cumplieran únicamente el objetivo de ruteo y transporte. En este caso, el diagrama representa la conexión punto a punto de un servicio identificado por un número de instalación. Sin embargo, como se ha mencionado, desde el PE central existen normalmente VRFs que contienen varios servicios de un mismo suscriptor, convirtiéndose en servicio punto-multipunto. En ese sentido, el entendimiento del diagrama no debe limitar a considerar un monitoreo punto a punto únicamente, sino que es la representación del monitoreo de un servicio en particular, sabiendo que las conexiones del equipo de monitoreo hacia las VRF del PE central pueden ser aprovechadas para el monitoreo de más servicios asociados. La línea continúa (en negro) representa la conectividad entre los equipos de la red que transportan el servicio. La línea roja representa la conectividad vía ping a la sede central y el ruteo realizado hacia la red a través del PE central. La línea azul en cambio, representa únicamente la conectividad

vía ping. No debe considerarse en ningún momento una conectividad directa entre el equipo de monitoreo y el PE remoto, ya que como se mencionó, el ruteo a la red se hizo únicamente a través del PE central. Debe considerarse en cambio, como una representación lógica de conectividad al PE y sede (CE) remota del suscriptor pasando por dicho PE.

El sistema de monitoreo MPLS IP-VPN se basó en configurar una sonda RPM con pruebas de RPM que tuvieran como destino, la IP de cada CPE en las sedes asociadas del suscriptor, incluyendo en ese sentido el CPE denominado como central. Es llamado de esa forma dado que representa la sede central del mencionado suscriptor y se encuentra ubicado del lado del PE al cual se realiza el ruteo para el RPM. Cabe destacar que con realizar el ruteo respectivo entre el equipo de monitoreo y el PE en mención, bastaba para que el ping del RPM alcanzara a los CPE respectivos, como se muestra en la Figura 67. La investigación de cada uno de los servicios fue fundamental en asegurar un monitoreo de punto inicial a punto final y de esa forma garantizar el monitoreo de nivel al cubrir toda la red del proveedor. Como complemento importante, la generación de gráficas en Cacti tuvo la importancia de reflejar resultados de monitoreo fáciles de interpretar a simple vista y en tiempo real, al alcance inmediato de los operadores por medio de pantallas de monitoreo. La generación de registros temporales e históricos por medio de Syslog permitió tener registros detallados que facilitaron aun más el proceso de diagnóstico. Esto principalmente, resultó siendo de gran valor. La identificación de los servicios IP a través de las descripciones de las VRF y subinterfaces centrales ayuda al operador a identificar de forma más efectiva un servicio, disminuyendo el tiempo de diagnóstico. La nomenclatura implementada, además dar un orden y fácil identificación de las sondas RPM, facilitó el proceso de búsqueda y organización de la información dentro del portal de servicios.

El primer paso en la implementación del monitoreo MPLS IP-VPN fue la búsqueda de documentación de los servicios IP, con el fin de encontrar las IP correctas para el monitoreo de punto a punto. El realizar esto de forma incorrecta implicaba un mal uso del monitoreo, pues el fin fue detectar caídas en cualquier punto en la red del proveedor. Así por ejemplo, si se monitorea la IP de la interfaz de salida del PE remoto, pero el servicio aún atravieza una red metro a nivel local, los eventos en dicha red nunca se identificarían y de hecho en ningún momento se tendría registro de las caídas del servicio provocadas por eventos en dicho tramo. Los diagramas del enlace resultaron siendo una ayuda inicial muy valiosa, ya que dieron una panorámica visual de la topología del servicio. Sin embargo, en ningún momento se consideraron como una herramienta definitiva en la búsqueda de las direcciones IP remotas. A veces, faltos de información sobre este tema y muchas otras veces, con información incorrecta, como se pudo determinar haciendo uso de otras formas de búsqueda. La documentación de bienvenida pudo resultar útil en función del tipo de información que presentara. Las pruebas de aceptación del servicio e imágenes de configuración conformaron el tipo de ayuda más valiosa. Sin embargo, ninguna fue considerada una prueba definitiva aunque las sesiones telnet confirmaran la conectividad a un CPE, debido a que por ejemplo, si una VRF contaba con 5 servicios, no se aseguraba que la dirección IP fuera exactamente del servicio en proceso de investigación. Fue frecuente, que en realidad la IP estuviera asociada a otro servicio dentro de la misma VRF, es decir, del mismo suscriptor. Una forma definitiva fue la búsqueda por gestión de equipos PE y de red local, por los que pasara el servicio. Como lo muestra la Figura 73, al observar la configuración de un PE remoto se puede determinar si la conexión a nivel de equipos de capa 3 es directa o no al CPE de la sede remota. De no serlo y si la gestión lo permitiese, el ir de salto en salto hasta llegar al equipo de acceso a la última milla y observar la configuración de la subinterfaz y protocolos de ruteo de acceso a la última milla resultó sin duda, una búsqueda definitiva de la dirección IP a monitorear. Un problema considerable con este método es que no siempre existió la

gestión de equipos en redes de operaciones locales. La gestión de los PE centrales, en cambio, estuvo siempre disponible y resultó necesario para cada servicio, determinar su VRF central y así obtener información del monitoreo a su sede central respectiva. El monitoreo a la central y al punto remoto fue la forma de realizar el monitoreo de punto a punto. El sistema de incidencias pasadas y actuales de los servicios (sistema de tickets) fue bastante efectiva, a pesar de tener la clara desventaja de un error humano en la documentación de revisiones. Para ello, se analizó a fondo cada caso cuando el uso de esta forma de búsqueda resultara necesario. El haber dos o más casos con las mismas revisiones de últimas millas, garantizaba con una mayor seguridad, el encontrar la dirección IP remota deseada. Las copias de configuración de dispositivos PE remotos o de red local no fueron muy útiles, ya que algunas de ellas estaban desactualizadas y en otros casos, mostraban conectividad IP a otros equipos core antes de llegar a la sede final. En pocos casos fue necesario utilizar el último recurso, consultas a operaciones locales sobre los servicios. Sin embargo, fue siempre efectivo, dado que se solicitó el apoyo tal que se brindase información sobre la configuración de la interfaz lógica, VRF y/o estado de la sesión BGP de acceso a la última milla. Ningún método por si sólo se confirmó como efectivo sin haber realizado las respectivas pruebas de ping y de telnet. La conectividad vía ping resultó necesaria, puesto que el servicio RPM utilizado fue del tipo *ICMP-PING*. Sin dicha conectividad el monitoreo no hubiera sido posible. La conectividad telnet fue una forma determinante (pero no siempre disponible), al asegurar la conectividad a algún CPE. Muchos suscriptores aplican listas de acceso o filtros que impidieron este tipo de conectividad. Como se mencionó, se encontraron servicios que compartieron la misma VRF central, pudiendo mezclar las direcciones IP entre ellos. Esto le dio incluso a telnet cierto margen de error. El complemento ideal en estos casos fueron la gestión de equipos remotos (cuando estuviera disponible) y muchas veces los reportes de incidencias sobre servicios. La Figura 145, por otro lado, muestra un avance regular (promedio de 30 a 35) servicios monitoreados por semana. Esto se debe a las facilidades y disponibilidad presentadas por las herramientas de monitoreo. Fueron necesarias alrededor de 8 semanas, como se puede observar en dicha figura, para cumplir con un poco más (253) de la cantidad propuesta (200) de servicios monitoreados.

La aplicación de los servicios RPM fue el segundo paso en el monitoreo MPLS IP-VPN. Antes de la configuración de sondas se estableció el ruteo por medio de la conectividad entre el equipo de monitoreo y los PE centrales. Una valiosa ayuda fue el hecho que las conexiones físicas entre los equipos PE y el equipo de monitoreo ya existían, haciendo necesaria sólo la conexión lógica. La virtualización de las subinterfaces del equipo de monitoreo en instancias del tipo Virtual-Router cumplieron el objetivo, como su nombre lo indica, de virtualizar cada conexión, evitando problemas de duplicación de IPs y que cada instancia tuviera su propia tabla de rutas. Para el caso del PE, la subinterfaz de conexión fue establecida dentro de la VRF específica del servicio. El direccionamiento de las redes de conectividad utilizadas fue escogido con base en que no fue usual observar su uso en algún otro punto del servicio. De todas formas, previo a su uso se revisaron coincidencias en la tabla de rutas de cada uno. Es importante mencionar que para el ruteo como tal, se realizó una ruta por default con un *next-hop,* igual a la dirección IP de la subinterfaz de conexión del PE. Esto indicó que para cualquier destino, se utilizara esa salida. Una vez la subinterfaz de conexión del PE estuviera introducida a la VRF, se aseguraba la conectividad hasta los dispositivos remotos a través de la red. Esto debido a que el *route-distinguisher* ayudó a realizar el ruteo dentro de la misma. Una vez establecido el ruteo y la instancia fuente en el equipo de monitoreo desde la cual se ejecutaría la conectividad ping, fue necesario probar dicha conectividad. En el caso de existir problemas punto a punto, se realizaban pruebas en la red de conectividad y también hacia las IP de las subinterfaces de salida de los PE. Los errores más comunes fueron el uso de distintas VLANs en las interfaces de conectividad, interfaz incorrecta aplicada en las instancias, o direcciones IP fuera de rango. Cabe

mencionar que la VLAN utilizada para la conectividad pudo o no ser la misma que la utilizada por la interfaz de conexión a la sede central dentro de la VRF del PE. Esto se diferencia del caso de las VPLS, como se verá más adelante.

La configuración de sondas RPM tuvo dos aspectos resaltantes, los estándares aplicados según la criticidad del caso, y la nomenclatura utilizada. El uso de *apply-groups* permitió desarrollar un tipo de script con el fin de ahorrar líneas de configuración sobre parámetros regulares para todos los servicios. Para cada estándar se desarrolló un *apply-group*. Tres estándares fueron definididos y básicamente diferenciados por frecuencia de sondeo. El primer estándar (definido en la Figura 82) fue aplicado en la jerarquía *[edit services rpm]* lo que indica que fue aplicado a todas las sondas configurables en el equipo. Esto debido a que es el estándar por defecto cuando un servicio no está en estado crítico. Para los servicios que fuera necesario aplicar otro estándar, el mismo simplemente se aplicó en jerarquías más específicas (*[edit services rpm probe], [edit services rpm probe <*> test])*, dependiendo si se deseaba aplicar el estándar sobre toda una sonda en particular o sólo sobre alguna de sus pruebas RPM. Junos OS se encarga de sobreescribir los parámetros repetidos, tomando en cuenta los definidios en jerarquías más específicas de primero. Es por ello que el primer estándar tiene más parámetros definidos que los otros, los cuales sólo necesitan especificar los parámetros diferenciadores. Así, el estándar 1 define el tipo de sonda RPM (*ICMP-PING)*, la cantidad de pines por prueba de RPM (15), el espacio de tiempo entre cada ping (30 segundos) y el espacio de tiempo entre cada prueba de RPM (150) segundos. Esto indica que una prueba de RPM se completaría en 450 segundos y la diferencia entre el inicio de una y otra daría un total de 600 segundos, es decir, 10 minutos. Esto fue pensado de esa forma ya que el portal de servicios realiza una actualización de la última prueba de RPM hecha precisamente cada 10 minutos, evitando perder información y sincronización si la frecuencia de sondeo fuera distinta. En el caso de servicios que usaran los otros estándares, esto quedó en un segundo plano, considerando que uno de los objetivos principales del portal es la detección pronta de incidentes. Una vez detectado, el tema de sincronización y obtención de la información completa en dicho portal no se considera priroritario, tanto como el hecho de estar al pendiente del servicio crítico, sin mencionar que esa información que se pierde del portal puede siempre extraerse de los equipos. Siguiendo con el estándar 1, se tomó el parámetro de peso de 128 bytes ya definido y este no varió de estándar en estándar. La razón fue que el objetivo primoridal del sistema de monitoreo es la detección de incidentes (es decir caídas) de los servicios y no tanto otros problemas como pérdida de paquetes, saturación o malconfiguraciones de MTU. Por último los umbrales configurados fueron tanto para la detección de un ping perdido (*succesive-loss=1)* como para el registro de una prueba total fallida *(total-loss=15)*. El parámetro de pines perdidos consecutivamente viene por defecto en 1, por lo que su configuración no fue del todo necesaria. Simplemente se enfatizó considerando que gracias al script, sólo se tuvo que configurar una vez. En el caso de prueba total fallida se configuró en 15, ya que una prueba RPM se compuso de 15 pines. La generación de logs que indiquen pruebas fallidas completas facilitan al usuario saber que el servicio está completamente caído. El estándar 2 (Figura 83) tuvo como parámetros diferenciadores, el tiempo entre cada ping (10 segundos) y el espacio de tiempo entre cada prueba de RPM (15 segundos). Se habla de un total de 165 segundos entre una prueba de RPM y otra. En ese sentido, el portal de servicios perdería dos pruebas completas y una parcial (4 pruebas de las cuales, obtendría únicamente la tercera, ya que sería la última en completarse). Por ello la importancia de sincronizar los tiempos cuando un servicio no es crítico. Por último, el estándar 3 fue el utilizado en casos en que el servicio necesitó de un constante monitoreo con el fin de determinar las causas del problema cuando estas no fueran obvias. Esta, además de diferenciar por tiempo entre cada ping (1 segundo) y entre cada prueba de RPM (1 segundo), también modificó la cantidad de pines por prueba (5). Esto, tal y como muestra la figura 6.18, obligó

a la modificación del umbral de pérdidas totales (*total-loss*) en 5. El uso de la nomenclatura aplicada facilitó identificar y ubicar los servicos RPM. Sin embargo, la aplicación estricta de esta nomenclatura (mostrada en el diseño experimental) buscó reflejar los resultados motrados del portal de servicios. A nivel de la jerarquía de configuración de servicio RPM, *[edit services rpm]*, el carácter "*c*" le permitió al portal identificar y clasificarlas por código de cliente, así como se muestra en la sección de resultados. Esto a nivel de visualización fue importante, ya que permite presentar el monitoreo en dicho portal de una forma resumida al operador, quien al ver alarmado alguna de las clasificaciones por código, sabe que dentro de dicha clasificación, podría haber una prueba RPM (de un servicio o instalación) dentro de una sonda RPM con inconvenientes. La clave en esta nomenclatura fue anteponer el carácter "*c*", seguido del código del suscriptor y el nombre de la VRF a la cual la sonda hace referencia. De esta forma se combinaron los intereses tanto para el portal como para la fácil identificación de las VRFs con RPM. Por otro lado, a nivel configuración de sondas RPM, *[edit service rpm probe]*, los caracteres "*c"* e "*i*" le permitieron identificar al portal las pruebas hacia sedes centrales y remotas. Esto fue importante para asociar todas las sedes remotas de un suscriptor a su respectiva sede central. Para las pruebas de RPM hacia la sede central, el caracter estuvo acompañado del código de cliente y su nombre. Para las pruebas hacia las sedes remotas, el carácter fue seguido del número de instalación. Esto para contribuir a la rápida identificación de servicios monitoreados. A nivel de jerarquía de pruebas de RPM se agregaron comentarios con identificación del *vrf-target*. El fin de esto fue identificar cada una de las pruebas, como instalaciones y centrales asociadas a un mismo suscriptor. De esa forma, habilitar un espacio en el portal en el que se viera un listado de instalaciones asociadas a una misma VRF. Por último la identificación de subinterfaces centrales con el carácter "*c*" antepuesto permitió al portal dentro de la jerarquía *[edit interfaces description]* asociar las distintas subinterfaces centrales por código de suscriptor, creando un espacio específico para ello en el portal.

Un complemento importante en el sistema de monitoreo MPLS IP-VPN, fue la generación de gráficas por medio de un gestor especilizado. En este caso se escogió Cacti por la facilidad que presenta en dicha generar dichos gráficos. La idea de utilizar esta plataforma es que una representación visual del estado de un servicio es mucho más fácil y rápida de entender para un operador, a través de pantallas de monitoreo. Además es una representación del estado actual e histórico del monitoreo de un servicio RPM. El script utilizado ya estaba disponible para su uso, y graficó dos parámetros del RPM únicamente, RTT y Jitter. El RTT es el más importante pues además de dar una idea de los tiempos de respuesta, su principal uso fue identificar la caída de un servicio, al dejar de graficar el RTT por la falta de conectividad. Aunque lo mismo ocurre con el Jitter, fue definido como un parámetro secundario que reflejó la dispersión de los tiempos de respuesta dentro de una prueba RPM. La búsqueda por número de instalación en la plataforma, agilizó considerablemente el diagnóstico de un incidente actual o reciente. Se debe considerar que la principal desventaja de Cacti, radica en las gráficas históricas por su frecuencia del poleo. Así por ejemplo, si se desean ver eventos de una hora específica 6 meses atrás, debe tomarse en cuenta que esto sólo puede ser observado a través del acercamiento de la gráfica RPM anual, cuya frecuencia de poleo es de cada 24 horas. Dada la definición de esta frecuencia, no sería posible observar el comportamiento deseado, ya que se observaría únicamente el RTT y Jitter promedio graficado durante todo ese día. Aun así estas gráficas son de utilidad en el sentido de visualizar el comportamiento histórico promedio e identificar eventos mayores, como caída de un servicio por varios días, por ejemplo. El complemento syslog en cambio, cubre esta desventaja presentada por las gráficas de RPM. Esto debido a que el registro histórico de logs es uniforme a lo largo del tiempo a diferencia de la frecuencia de poleo de las gráficas. Syslog se encarga de generar un log y guardarlo en archivos generados o servidores especializados, identificando tipo, duración, lugar, fecha y hora

de un evento presentado 6 meses atrás, por ejemplo. Es por ello que Syslog se consideró una herramienta aún más importante que Cacti en el proceso de diagnóstico de un incidente. No así, en el sentido de alerta al operador, en el que la fácil y rápida visualzación de gráficos hace a Cacti preferible en ese sentido, considerando que el para el operador es más simple y eficiente de esa forma que estando atento a logs de servidores y archivos.

La configuración de archivos también utilizó una nomenclatura definida, en este caso, para la identificación agilizada de los mismos. Únicamente se antepuso la secuencia *"rpm_"*, seguido del nombre de la sonda RPM para la cual se generarían los eventos especificados. Los niveles de recurso y severidad no fueron trascendentes en este caso ya que no interesó la clasificación de los mismos en algún nivel específico, tanto como la generación de eventos configurados. Tal y como se especificó en la configuración de sondas RPM, fue necesario definir coincidencias que mapearan a los umbrales definidos en las mismas. Así el evento denominado como "*PING-PROBE-FAILED"* se genera cuando se supera el umbral de pérdidas de ping consecutivas (en este caso, parámetro fijado a 1); mientras que el "*PING-TEST-FAILED"*, cuando se supera el umbral de pérdidas totales (fijado a 15). Es importante resaltar que acompañado a la especificación de estos eventos, se acompaña el nombre de la sonda RPM de la cual se desea identificar dichos eventos. La Figura 43 detalla el mensaje para cada uno de estos eventos y se debe mencionar que el *match* utilizado para especificar eventos de sondas específicas debe ir tal y como se detalla en la mencionada figura, que es cómo se aplicó según se muestra en la Figura 91. En la misma se observa la coincidencia deseada, sin espacios faltantes ni sobrantes, ni el uso aleatorio de mayúsculas y minúsculas. En cuanto a la elección de los parámetros de tamaño y número de archivos, esto se detalla en la sección 4.1, "Syslog por generación de archivos", en el diseño experimental. Los puntos clave para esta definición fueron el espacio disponible en el directorio de almacenamiento, y el peso de un log para definir al cantidad de logs total que se deseó almacenar de forma temporal para cada servicio. Se tuvo disponibles 22GB (71%) de la capacidad total del directorio y se estimó en 1KB el peso de un log. En ese sentido se decidió definir en 10 la cantidad de registros por archivo, con un tamaño de 1MB. De esa forma, se aseguró una capacidad de 1000 logs por registro de archivo y un total de (10MB) 10,000 logs por archivo completo para un servicio especifico. Tomando en cuenta que son 200 servicios los planeados a monitorear, la capacidad total ascendiería a 2GB de 22GB disponibles. Esto, al representar únicamente el uso de un 5% del directorio y tomando en cuenta la importancia de syslog para el monitoreo de servicios IP, se consideró bastante aceptable. Así, por ejemplo si un archivo cubre una sonda de 10 pruebas RPM, el archivo representa 10 instalaciones o servicios distintos. Por lo tanto el tamaño en ese caso se definiría a 10MB. El envío de logs al servidor especializado, aseguró el registro de los archivos de forma permanente. Se debe resltar que permanente en el sentido que duraría mucho más tiempo que el registro de syslog por archivos. En la Figura 164 se pueden observar los resultados de syslog al generar eventos. El hecho que se hayan generado logs para perdidas de pines individuales ayuda a cubrir la mayor parte de eventos posible. Esto no se hubiese logrado únicamente con la generación de logs por pérdida total de prueba RPM. Por ejemplo, si un servicio sufiera una caída por 7 minutos (la prueba RPM está definida para hacerse en 7 minutos y 30 segundos, a lo que luego se le suma la espera de 2 minutos y 30 segundos), el log de pérdida total no se generaría, mientras que sí se generarían 14 logs de pruebas de ping individuales perdidas. En esto radica la importancia de un registro de eventos por ping perdido. Sin embargo, los logs de pérdidas completas de pruebas de RPM cumplen con el fin de informar de forma más ágil al operador de que un servicio lleva caido un tiempo considerable. Es más sencillo para el operador al momento de realizar un diagnóstico, filtrar por logs de pérdidas completas que revisar uno a uno los logs de pérdidas individuales. Por lo tanto, ambos tipos de logs tuvieron su fin en colaborar con permitir la elaboracíon de un diagnóstico de nivel. Finalmente, se optó únicamente

por Syslog por envío de eventos a un servidor Syslog al que los operadores en turno tuvieran acceso como la opción regular para el almacenamiento de eventos. Como se ve en la Figura 165, a dicho servidor se sube practicamente cualquier tipo de log, incluyendo syslog para eventos de RPM, teniendo únicamente que filtrar por el tipo de interés. Syslog por generación de archivos se dejó como una opción viable únicamente para casos críticos. Esto debido a que se consideró un desperdicio de recursos el hecho de generar syslog por archivos cuando ya exisitía la opción de syslog por envío a servidor.

Por último, la identificación de VRF y subinterfaces centrales cumplió con el fin de proveer una fácil identificación de los servicios IP. Ambas tuvieron la misma descripción para cada VRF asociada a un PE central, con la clara excepción que las subinterfaces centrales llevaron una nomenclatura específica utilizada por el portal, lo cual fue abordado con anterioridad en esta discusión. El hecho de colocar el nombre del suscriptor y todas sus instalaciones asociadas por país fue clave en facilitar la búsqueda de dichos servicios tal y como puede verse en la figura 166. Cabe resaltar la importancia de la canalización | *display set*, ya que muestra las coincidencias en formato de configuración, lo que permite observar sobre qué jerarquías se posicionan las coincidencias y en consecuencia, agiliza la búsqueda. Esto no ocurre al utilizar sólo la canalización | *match,* como puede observarse en la mencionada figura.

El monitoreo MPLS VPLS tuvo dos pilares, la generación de gráficas de tráfico y la implementación de un algoritmo de diagnóstico inicial de los servicios basado en el reconocimiento de direciones MAC de las tablas de de dichos servicios. Se debe resaltar que el fin no fue realizar un diagnóstico a detalle, sino uno que fuera suficiente para determinar un servicio como operativo o no operativo. Además, como complementos importantes estuvo la correcta identificación de VPLS y subinterfaces centrales y una forma alternativa de monitoreo, haciendo uso del sistema implementado en servicios MPLS IP-VPN. Un servicio VPLS asegura la conectividad entre sus sitios a través del estado actual de la conexión. Sin embargo, en este caso el término “sitios” debe asociarse a los PE implicados dentro de la red MPLS. Esto da a entender que si bien una conexión VPLS puede estar operativa, esto no garantiza en ningún momento que el servicio hacia las sedes remotas del suscriptor estén funcionando. Por ejemplo, una instalación que luego de la red MPLS internacional pasa por una red Metro local, puede bien no tener falla a nivel de red MPLS y por ello mostrar las conexiones VPLS sin problemas, pero pueden existir incidentes en la red local que dejen fuera esa instalación en particular. Es por ello que el monitoreo de las conexiones de un servicio VPLS se descartó como posible forma de monitoreo. Al ser para el sucriptor un servicio VPLS transparente y entendiendo que las MAC asociadas al servicio son reconocidas por el mismo al levantar una conectividad IP y generar tráfico, una forma de implementar parte de este monitoreo fue la evaluación de las tablas MAC de las VPLS. Ambas razones llevan a concluir que las direcciones MAC reconocidas deben ser en el mayor de los casos del suscriptor y no del proveedor. Las excepciones más comunes son asociadas a monitoreo de nivel IP por parte del proveedor, pero para ello, se desarrolló un algoritmo de nivel, por medio del cual no sólo se identificaron direcciones MAC, sino que se compararon con respecto a un registro histórico de direcciones obtenido al momento de considerar el servicio como operativo. El diagnóstico automatizado de un servicio VPLS entonces, se basó en la implementación de un algoritmo de evaluación de disponibilidad y posibles cambios de las direcciones MAC en su tabla VPLS. Lamentablemente, las interfaces LSI existen en función de los sitios PE y no de las interfaces lógicas de cada sede asociada a cada sitio. De esta forma, no fue posible determinar desde los PE centrales la cantidad total de sedes remotas, sino que sólo de sitios PE. Esto fue una limitante considerable para el diagnóstico elaborado, pues un PE remoto podría tener varias sedes asociadas y con recibir una dirección MAC de una de ellas fue suficiente para

considerarlo operativo, cuando en realidad el resto estarían fuera de operación. Con el registro histórico de direcciones MAC y sus cantidades por VPLS, se buscó compensar esta limitante. Esto debido a que a una VPLS se le asignaba un registro histórico sólo si la misma no se detectaba fuera de operación a través de la primera parte del diagnóstico inicial. La segunda herramienta de monitoreo fue el tráfico del servicio. Se consideró que si para un servicio VPLS se detectaba una suficiente cantidad de tráfico a través de las subinterfaces de salida de la red MPLS de los PE, no era necesario realizar la evaluación del algoritmo, y he ahí su importancia, a pesar de lo simple que resultó su implementación. La correcta identificación de los servicios tuvo exactamente el mismo fin que la aplicada en servicios IP. Se determinó que es factible implementar el sistema de monitoreo de RPM para los servicios VPLS, haciendo pruebas exitosas. Sin embargo, por ser un servicio transparente, es necesario el apoyo del suscriptor, autorizando, configurando y brindado una red de monitoreo. Como se explica más adelante, esta opción no resultó eficaz ni escalable, aunque sí más eficiente que el monitoreo propuesto para servicios VPLS.

El sistema de monitoreo MPLS-VPLS se describe a un alto nivel en la Figura 99. En la misma se reflejan los dos pilares previamente mencionados, observando como de las VPLS configuradas en los PE asociados, se obtiene tanto direcciones MAC como tráfico. Difiere del diagrama IP (Figura 67) en el sentido que no es necesaria una conexión directa hacia los CE, ni siquiera a los equipos de red local, sino que únicamente a los sitios (PE) de la conexión VPLS. Esto debido a que únicamente es necesaria la conexión a los mismos para obtener las direcciones MAC y tráfico asociados. De hecho, debido a que los PE (como los puertos de un switch) comparten la misma tabla MAC, bastó con la conexión hacia uno de los PE para el tema de detección de MAC. En este caso resulta indiferente el PE seleccionado (no debe ser el PE central como en el caso de los servicios IP), sin embargo por temas de gestión, ambos PE centrales por los que pasan los servicios VPLS, fueron nuevamente seleccionados. Al igual que para el diagrama IP, el entendimiento del diagrama no debe sugerir la estricta posibilidad de que un enlace VPLS es punto a punto. Un enlace VPLS, al igual que una VRF puede resultar siendo punto-multipunto. El diagrama de la Figura 99 es la representación del monitoreo de un servicio o instalación en particular, teniendo claro la posibilidad de que hayan más servicios involucrados en la misma VPLS. Debe ser notoria la representación de una laptop en lugar de un router como equipo de monitoreo, debido a que, al no realizarse acciones de conectividad vía ping ni ruteo, un dispositivo de capa 3 no resultó necesario ni efectivo para este fin. De hecho es desde un ordenador que se ejecutaron las acciones de generación de tráfico e implementación del algoritmo para la revisión del PE central. El dispositivo de monitoreo, al igual que en capa 3, fue utilizado desde fuera de la red, con el fin de dejar en los equipos Core y PE solamente las tareas de transporte y configuración de servicios. La línea en color café representa la conectividad de transporte de servicios, mientras que la azul, la conectividad a los equipos para la extracción de la información necesaria. Una vez más, la línea de conectividad al PE remoto no se debe entender como una conexión directa en el caso MAC, sino como una representacion de que a través de le conexión al PE central, se obtienen las direcciones MAC locales del PE remoto. En el caso de la extracción de tráfico, sí se puede ver como una conexión directa, por temas de gestión SNMP, al momento de graficar en los PE remotos.

La investigación de las herramientas de monitoreo fue el primer paso a llevar a cabo para la implementación de este sistema de monitoreo. Esto, una vez que se tuvo claro que realizar el monitoreo RPM no era una opción escalable por depender del suscriptor, y una vez se fijó como métodos del sistema, el diagnóstico por reconocimiento de direcciones MAC y las gráficas de tráfico. SNMP fue una de las primeras opciones a considerar, ya que resulta siendo un protocolo ampliamente utilizado para el monitoreo de características de equipos de red. Además, su uso es

simple al no requerir levantar ningún tipo de conexión ni autenticación especial con los mismos. Esto conlleva una ventaja especial debido a que evita un consumo adicional del procesamiento de los equipos a monitorear. SNMP se sirve únicamente de OIDs para identificar los objetos a monitorear. El gran problema que se encontró con este protocolo es que las direcciones MAC de las tablas VPLS no tienen OID asociado. Esto hasta cierto sentido es lógico, dado que dichas direcciones MAC no representan un componente físico o lógico como tal del equipo del que se desea extraer la información. Por el contrario, son direcciones obtenidas de equipos remotos que se propagan a través de la red WAN del proveedor. Aun así, existen OIDs que identifican direcciones MAC reconocidas por ARP, que por tanto, tampoco son del equipo como tal, aunque sí del mismo segmento de red de la interfaz asociada al equipo donde se reconoce la MAC. Lo que se hizo para determinar la no existencia de OIDs en MACs de tablas VPLS fue realizar un barrido del equipo tal y como se muestra en el diseño experimental y se ve reflejado a traves de las figuras 6.34, 6.35, 6.36 y 6.37. Las pruebas se realizaron con dos búsquedas, las de direcciones MAC de interfaces físicas del equipo y las de direcciones MAC asociadas a servicios VPLS. Si bien pudo usarse la búsqueda a partir de la jerarquía SNMP 1.3.6, el hecho de hacerlo desde la 1, fue simplemente para cubrir todas las posibilidades. Como se pudo apreciar en dichas imágenes, existe un OID definido para cada interfaz física de un equipo, el cual es también asignado a sus subinterfaces. Sin embargo, al aplicar el filtro para una MAC de una VPLS, el equipo tarda varios minutos previo a finalizar la búsqueda sin resultado alguno. Las pruebas se hicieron con direcciones MAC de interfaces físicas con el fin de justificar la búsqueda de la forma: “AA BB CC DD EE FF”, ya que así es como lo requieren los equipos PE. Debe notarse que se utilizan un espacio en lugar del carácter “:” y que luego de los primeros 32 bits u ocho hexadecimales, se debe colocar un doble espacio. Se concluyó que a pesar de las facilidades mencionadas, SNMP no podía ser tomado en cuenta como una opción si la implementación del diagnóstico de VPLS debía estar basada en la detección de direcciones MAC. En cambio, las pruebas realizadas por medio de NETCONF, luego de la instalación de PyEZ y sus dependencias demuestran que por medio de RPCs definidos dentro del Junos XML API, es posible obtener cualquier información del equipo, incluyendo en este caso, tablas MAC y estado de las conexiones de una VPLS. La principal desventaja de NETCONF es que cada vez que se corre un programa, debe iniciarse una sesión confiable y una autenticación por medio del usuario utilizado. Esto tiende a consumir tiempo y recursos del equipo. Por ello, se decidió definir a través de las gráficas de tráfico, cuando o no correr el programa de diagnóstico. Esto se profundiza más adelante. Cacti, debido a las facilidades que presenta para realizar gráficas de tráfico, fue seleccionado como gestor para este fin. El objetivo fue obtener las gráficas provenientes de las interfaces de salida de los equipos PE de la red MPLS. Para ello, fue necesario requerir lectura SNMP de los equipos implicados a las operaciones locales de cada país asociado.

Luego de haber comprendido el funcionamiento de una VPLS y tener definido cómo utilizar las herramientas de monitoreo para obtener la información necesaria del PE central, el siguiente paso fue definir e implementar el algoritmo. Este algoritmo se compuso de dos partes, una para obtener y realizar el parseo de la información necesaria, y la otra para realizar el diagnóstico con base en dicha información. Tres librerías de Python fueron utilizadas. *JNPR-JUNOS* por medio del objeto *DEVICE* sirvió para realizar los RPCs al servidor NETCONF del equipo. *LXML* por medio de *ETREE* para realizar la conversión de formato XML a STRING. Por último, *SYS* para realizar una salida segura del programa cómo método de excepción al encontrar errores en el establecimiento inicial de la sesión y la autenticación. El primer paso previo a definición de parseo y diagnóstico fue determinar qué información útil obtener para cumplir con el objetivo y a través de qué comandos realizarlo. Los comandos que se utilizaron fueron: “*show vpls mac-table”, “show vpls connections extensive”* y *“show configuration vpls”*, especificando el sistema lógico de los servicios para obtener la

información de todos los servicios por RPC al ejecutarlo desde el programa. En este punto cabe destacar que para el comando de la configuración no se utilizó un RPC, pues como la Figura 111b muestra el mismo no se encontró disponible para este comando. Esto obligó a utilizar la función CLI de la clase DEVICE y parsear la información en formato ASCII. Como se explicó en el marco teórico esto representa una desventaja pues los cambios en la presentación de información que se pueden dar a través de diferentes versiones del JUNOS OS no hace al formato ASCII escalable. Sin embargo se tuvo que lidiar con este inconveniente, con la ventaja que este comando de configuración sólo se utliza sólo en un caso muy especial (cuando existen dos o más interfaces locales asociadas al sitio local, lo cual no es muy común en VPLS de los PE centrales). En cambio, el formato XML maneja una estructura que se mantiene a través de las distintas versiones del JUNOS OS. Por las ventajas que presenta el formato XML sobre el ASCII convencional (más detalladas en el maco teórico), se utilizaron solicitudes RPC para obtener los datos de los otros dos comandos (conexiones y tablas MAC de VPLS). El comando de las tablas MAC de las VPLS fue el principal y el más necesario por que como el nombre del mismo lo sugiere, su función es proveer al operador de la tablas MAC. El comando es útil en su modo extendido porque a través del mismo se puede determinar la cantidad de sitios, interfaces lógicas locales y remotas. En este caso como ya se ha explicado, el número de sitios es igual al número de interfaces remotas LSI. Es decir, que para los sitios remotos no fue posible determinar la cantidad de sedes asociadas. Esto hubiera sido posible si por cada sede asociada a un sitio se recibiera una LSI distinta. Sin embargo, por ejemplo, si para un sitio PE remoto hay tres sedes, y una esta fuera, desde el PE central se seguiran recibiendo direcciones MAC de la LSI de dicho PE remoto (dado que las otras dos sedes sí estarían operativas), sin poder identificar la sede que está fuera. No fue así con las interfaces lógicas locales, de las cuales sí se hace diferenciación. (Lo anterior ha sido demostrado en el diseño experimental). Aun así, el hecho de no diferenciar la cantidad de sedes por sitio remoto representó una clara limitante para el algoritmo implementado. Como se mencionó anteriomente, el registro histórico buscó aliviar este inconveniente. A través del programa de detección y diagnóstico, se procedió a obtener o realizar el parseo de información sobre interfaces lógicas, y su debido análisis. Este comando de conexiones VPLS, resulta siendo un paso a realizar, previo al diagnóstico como tal, dado que es de vital importancia conocer la cantidad de sitios remotos e interfaces lógicas locales activas del servicio antes. El comando de la configuración fue necesario porque a través de la revisión de las conexiones no es posible determinar qué interfaces lógicas locales están activas y cuáles son de respaldo. Esto se comprobó en el diseño experimental, así como el hecho que las interfaces lógicas remotas LSI representa una limitante para el diagnóstico. Aun así, cabe mencionar que para los servicios de este ISP en particular, lo más normal actualmente es tener una sede por sitio. Del comando de configuración es importante detallar si se identificaba una interfaz local con el parámetro de activa primaria, el resto automáticamente quedaban desactivadas. Si el parámetro no se encontraba configurado sobre alguna de las interfaces locales, entonces todas se consideraban activas. Esto se comprobó en el diseño experimental, figuras 6.46b y 6.46c, al detectar direcciones MAC de todas las interfaces al no tener aplicado el parámetro, así como al sólo detectar de la interfaz seleccionada al sí aplicarlo. Para el comando de las conexiones, todas las interfaces locales se consideraban activas en cualquiera de estos dos escenarios. He ahí la necesidad de utilizar el comado de configuración. Este escenario fue muy particular, debido a que sólo se evaluó al detectar más de una interfaz lógica local en las conexiones de la VPLS en evaluación.

Los pasos del algoritmo y su debida implementación a través del uso de PyEZ y sus dependencias pueden verse en el diseño experimental. Los mismos están basados en el entendimiento de cómo funciona una VPLS y cómo un operador normalmente diagnostica un servicio, con ciertas mejoras de comparación de cantidad y direcciones MAC con respecto a registros históricos, con el fin de

determinar que para todos los sitios remotos y sedes locales, exista una dirección MAC asociada. Como parte del algoritmo, el primer paso fue obtener la información de las tablas MAC y conexiones de los servicios, separándolo en elementos de listas independientes. Luego en el ciclo general de revisión de tablas MAC, se realizaron las coincidencias por servicio para asociar la información de las interfaces lógicas y remotas obtenidas de las conexiones VPLS como parte del diagnóstico. A pesar de que la implementación del programa estuvo basada en el algoritmo, los pasos tal y como se explican en el diseño no son necesariamente los mismos en cantidad. De hecho hay pasos en la implementación conformados por más de un paso del algoritmo. El uso de los elementos tag mencionados en el diseño experimental, fue de gran utilidad para realizar el parseo de la información. Tal y como se mencionó en el marco teórico, los elementos tag se componen de un elemento de apertura y otro de cierre. Por esta razón, al momento de dividir las cadenas por medio de estos elementos, de un lado quedaba información insevible para los fines del programa, y del otro lado quedaban los datos buscados. Específicamente al dividir una cadena por medio de un tag element, en el índice 0 de la lista queda la información previa a la apertura del elemento tag, es decir información de poco valor. En el índice 1, quedan datos entre los elementos de apertura y cierre, es decir, la información buscada. En el índice 2, finalmente queda una vez más información sin utilidad debido a que la misma se presenta luego del elemento tag de cierre. Es por esa razón que se mencionó en varias ocasiones que los datos de valor siempre se ubicaron en los elementos impares de las listas utilizadas. Además, el hecho que los elementos tag se compongan de una llave de apertura y otra de cierre, obligó a dividir en dos la cantidades reales de los componentes buscados. Por ejemplo, si al determinar el número de direcciones MAC de una tabla, a nivel de CLI se ve que son seis, la lista se compondrá de doce elementos, ya que en ella queda almacenada la información de poca utilidad en los índices pares. Para obtener la cantidad real, se midieron las longitudes de las listas y se realizó la división por dos. Esto se dio al momento de determinar el número de conexiones VPLS, el número de tablas MAC de VPLS y el número de direcciones MAC por tabla de VPLS. Al contrario, al momento de la elaboración de ciclos para la búsqueda de servicios en los elementos de las listas, los mismos dependieron de la cantidad de componentes de interés multiplicada por dos, es decir de la cantidad total de los elementos de la lista. Esto porque las cantidades determinadas de componentes de interés se refieren únicamente a la información de utilidad ubicada en los índices impares, pero la duración de los ciclos tuvo que ir en proporción directa a la cantidad total de elementos de las listas para evitar errores en el programa. Como la cantidad de componentes de utilidad sólo considera los impares, fue necesaria la multiplicación por dos. Ya dentro del ciclo se estableció una condición de analizar la información únicamente de los componentes de interés (impares). En el diseño experimental se realizó una vasta descripción de los pasos del algoritmo y como se desarrolló para implementarlo utilizando NETCONF, JUNOS PyEZ y sus dependencias.

Una vez se tuvo el programa implementado y funcional al realizar pruebas sobre un equipo PE, se procedió a evaluar todos los escenarios posibles, producto del algoritmo. Esto con el fin de asegurar el comportamiento esperado y corregir las anomalías encontradas. Como puede verse en la sección de resultados, la presentación de la información se realizó de una forma amena al operador con el fin de hacer ágil el entendimiento de la misma. El primer caso fue comprobar la diferenciación que realiza el programa sobre servicios punto-punto y punto-multipunto que se puedan presentar. En otras palabras, se evaluó que la busqueda de conexiones para determinar la cantidad de interfaces lógicas activas funcionara. Esto incluyó la evaluación de casos en los que existiera más de una interfaz lógica local, separando la activa de las de respaldo cuando esto aplicara. En la Figura 168b, el programa ignora el resto de interfaces al detectar una como activa primaria. En la Figura 168c las considera todas al no detectar el parámetro sobre ninguna. El primer

caso da a entender sólo una sede asociada al sitio. El segundo da a entender varias sedes por sitio. Luego de este proceso, el siguiente escenario evaluó que el servicio fuera detectado correctamente operativo. Este análisis incluyo asegurar que se detectaran direcciones MAC provenientes de cada interfaz lógica activa, lo cual se dividió en dos pasos dentro del algoritmo. El primero determinar que la cantidad de direcciones MAC fuera igual o mayor al de las interfaces lógicas activas; y segundo, la evaluación como tal de recibir direcciones de cada una de dichas interfaces. Esa evaluación consitió en el análisis de las interfaces lógicas locales y remotas individualmente. Luego, se evaluaron los casos en los que cada una de estas condiciones no se cumplieran, observando que los resultados de diagnóstico del programa fueran los adecuados. Por el contrario, al cumplirse estas condiciones se consideró el servicio operativo, pues al detectar direcciones de cada punto final es suficiente para asegurar la conectividad a nivel de VPLS. Así, por ejemplo, en la Figura 168b, al recibir dirección MAC de la interfaz activa primaria únicamente, detecta el servicio operativo pues las otras se desactivan al especificar el parámetro de activa primaria sobre la seleccionada. Lo contrario ocurre en la Figura 168c, ya que al no detectar dicho parámetro utilizando el comando de configuración, las considera todas activas y espera recibir al menos una dirección MAC de alguna de ellas. En el caso de esta figura sí se cumplió el requisito y por ende se consideró operativo. El inconveniente encontrado realizando únicamente este análisis fue que en realidad no se tenía la certeza de dónde estaban ubicados los equipos detectados a través de sus direcciones MAC. Las direcciones MAC podrían pertenecer a cualquier equipo de la red, desde el equipo de monitoreo utilizado en capa 3 (en los casos que RPM fuera utilizado para VPLS) pasando por las redes locales de las operaciones hasta las sedes remotas del suscriptor, ya que en cualquier punto podría levantarse conectividad IP. Esto obligó a ir más alla de las revisiones convencionales realizadas por el operador, diseñando una segunda parte del algoritmo de diagnóstico basada en detectar cambios de direcciones MAC y cantidad de direcciones con respecto a registros históricos de los servicios, obtenidos en momentos en los que los mismos se detectan operativos. Lo primero fue llevar a cabo estos registros históricos. El fin de los mismos fue tener una comparación histórica disponible para lograr los resultados esperados por esta parte del diagnóstico. Cabe mencionar que para crear estos registros, el servicio en evaluación debía haberse considerado operativo por la primera parte del diagnóstico y la creación de los mismos no debía haberse realizado con anterioridad. Si el servicio no cumplía con estar operativo, entonces la asignación de variables de comparacíon se igualaba a las variables del barrido actual del servicio, haciendo que el resultado de esta segunda parte fuera siempre el de no detectar cambios de direcciones MAC ni de sus cantidades. En otras palabras, dejando nulo el efecto de esta parte del algoritmo. Esta solución fue implementada así, para obligar al programa a siempre realizar asignaciones y evitar errores en las comparaciones. Si únicamente se cumplía con ya haber realizado la creación de registro histórico, entonces se procedía a llevar a cabo el procedimiento. En esta parte del diagnóstico cabe destacar que el servicio en todo momento se consideró operativo pero con condiciones que deben ser tomadas en cuenta por el operador para proceder con su revisión. Esto lo diferenció claramente de la primera parte del mismo, en la que al no detectar direcciones MAC de algún punto local o remoto, efectivamente el servicio debía considerarse no operativo.

La implementación de esta parte requirió un cuidadoso uso de las funcionalidades de Python, como los dobles ciclos, condicionales y variables booelanas. Requirió también realizar pruebas manuales consecutivas sobre los equipos en búsqueda de todos los escenarios posibles según el servicio evaluado. Para ello se programó un ciclo infinito en el programa. Así, por ejemplo, en el caso de encontrar un servicio operativo pero con menos direcciones MAC de las esperadas acorde a los registros históricos, se revisó manual y minuciosamente la información presentada por el programa con respecto a casos anteriores, determinando qué dirección MAC faltaba y que las

condiciones para que el servicio se considerara operativo en efecto se cumplieran. En los escenarios de esta parte del algoritmo, se analizó el cumplimiento de las condiciones tanto de la primera como de la segunda parte del proceso de diagnóstico. Este tipo de acciones fueron las comúnmente realizadas para asegurar el correcto funcionamiento del programa. Uno de los escenarios clave en la toma de decisiones de actualización de registros fue el de encontrar el servicio operativo con más direcciones MAC de las esperadas pero con cambios en una o más direcciones MAC originales. En este caso en ningún momento se consideró la actualización de la información, porque cabía la posibilidad de, por ejemplo, haber perdido la única dirección MAC reconocida de alguna sede, a cambio de dos direcciones adicionales de otra sede, en la que originalmente ya se conocían direcciones. (Recibir varias direccciones MAC de una sede ocurre cuando el CE es un switch, como se puede recapitular del marco teórico, ya que los mismos no limitan el reconocimiento de direcciones como lo hace un router). En ese escenario particular, el servicio VPLS en realidad debería considerarse caído (tener una sede caída implica tener un servicio o instalación fuera) a pesar de reconocer más direcciones y ser considerado operativo por la primera parte del diagnóstico. Es por ello que para estos escenarios fue importante establecer una condición a la afirmación de servicio operativo y así, que el operador en turno lo considerara como una posibilidad de servicio fuera. El problema surge si inicialmente en el PE central, el servicio VPLS en el mencionado ejemplo se considera operativo al recibir direcciones MAC de la LSI (la misma para todas las sedes del sitio remoto) cuando en realidad el sitio remoto al que hace referencia la LSI, tiene una sede caída, y se graba así en los registros históricos. A pesar de que se ha mencionado que los registros buscaron aliviar esta limitante, se puede ver en este escenario particular, que los registros históricos no logran cubrir en su totalidad este punto débil del algoritmo. Otro escenario es el de detectar un servicio operativo, con misma cantidad de direcciones MAC pero cambios en alguna de ellas. En este, tampoco puede haber actualización de registros ya que se consideran las mismas posiblidades que el escenario anterior. Por último, el escenario final que se pudo esperar bajo la realización de este diagnóstico automatizado, fue el de detectar el servicio operativo, con más direcciones MAC y sin cambios en relación a las originales. En este caso, la actualización de registros se considero realizable y asi fue implementado en el programa. Esto implícitamente buscó corregir registros históricos incorrectos, pudiendo agregar direcciones MAC que aseguren la completa operatividad. El despliegue de mensajes de diagnóstico se realizó en función del resultado dado por condiciones y variables (sobre todo, booleanas), dentro de ciclos dobles y simples. De hecho, muchos de los casos que se debieron corregir por no encontrar los resultados esperados en el programa, provenían más de errores en el despliegue de mensajes que en la lógica empleada por medio de condiciones y variables. Por ello, se debió tener también un cuidado especial en la programación de despliegue de estos mensajes de diagnóstico.

La implementación de gráficas de tráfico fue de igual importancia que la implementación del algoritmo, por lo que no puede ser considerada como un complemento del sistema de monitoreo. A pesar de la complejidad en el desarrollo e implementación del algoritmo, en contraste con la simplicidad en la generación de estas gráficas, puede decirse que sin ellas, el algoritmo no sería de demasiada utilidad. La razón se debe a que, como ya se ha mencionado, NETCONF requiere del establecimiento de una sesión entre la aplicación cliente y servidor, añadido a una posterior autenticación. Durante las pruebas realizadas, se pudo comprobar que al equipo PE le tomaba alrededor de 10 segundos terminar de establecer la conectividad y tener acceso a la información del equipo. Esto incrementa el uso de los recursos del equipo; y dejar el programa corriendo infinitamente, evitando tener que pasar por estos procesos, tampoco fue una opción, pues el hecho de estar constamente activo sobre el equipo, ya era una demanda alta de sus recursos. Entonces, se decidió que el programa se corriera únicamente en los casos en los que se detectase algúna

interfaz PE sin tráfico y por ello la importancia que finalmente se le dio al monitoreo por consumo en los servicios VPLS. Sin embargo, también podría darse la posibilidad de que un PE remoto tuviera varias sedes asociadas y que al caerse una de ellas, se siguiera observando tráfico activo de la subinterfaz de dicho PE. Esto debido a que las otras sedes podrían seguir consumiendo el ancho de banda sin problemas. Dada esta posibilidad, el programa también decidió correrse automáticamente en diferentes momentos del día. Cabe destacar como pequeña ventaja, tanto en los sitios locales como remotos, actualmente para el ISP para el cual se desarrolló este trabajo, cuenta únicamente con una sede por sitio en los servicios VPLS que ofrece. La relación entre el tráfico y el algoritmo fue realizada a través del portal de servicios (realizado por terceros, como se ha mencionado), organizando los servicios por suscriptor (gracias a la identificación de VPLS y subinterfaces) y mostrando diferentes tipos de alarmas de acuerdo al estado (tráfico activo, tráfico no activo y los resultados posibles del algoritmo), de tal forma que visualmente para el operador de turno fuera fácil identificar un posible incidente y anteponerse al reporte del suscriptor. Para los fines de este trabajo en específico, únicamente se elaboraron las gráficas como tal, recibiendo la autorización de las operaciones locales para la lectura de tráfico SNMP. Esto debido a que Cacti utiliza dicho protocolo para obtener la información a través de MIB y las respectivas OIDs. Las gráficas tienen la mismas características que las de RPM, con la obvia diferencia que dibuja tráfico de entrada y salida de la subinterfaz en lugar de RTT y Jitter. En consecuencia, deben considerarse las mismas limitaciones para las gráficas históricas debido a la frecuencia del poleo. En cuanto al avance del monitoreo de VPLS, la gráfica de la Figura 179 muestra el progreso en función del tiempo que se tuvo con respecto a las gráficas de tráfico. El desarrollo del algoritmo no cuenta en este caso, ya que una vez realizado e implementado, el monitoreo sobre todos los servicios por equipo PE es inmediato.

Finalmente, como complemento se realizó la debida identificación de las VPLS y subinterfaces centrales para agilizar el proceso de diagnóstico; y desde el portal de servicios, asociar las centrales por suscriptor. En esencia, cumplió el mismo objetivo que la identificación realizada a nivel IP. Se propuso como un complemento extra la realización del monitoreo RPM aplicado a servicios VPLS. Los pasos básicamente fueron los mismos, sin embargo se encontraron desventajas importantes por lo que se descartó su implementación. Una diferencia que no tuvo impacto en la toma de esta decisión fue encontrada en el proceso de conectividad lógica. La subinterfaz de conexión en el PE debió ser configurada como de familia VPLS en lugar de familia IPv4. Esto debido a que la misma debía ser introducida dentro de la instancia VPLS para que la conectividad funcionara, y como era de esperarse, una instancia VPLS sólo admite subinterfaces de la misma familia. Otra diferencia en la conectividad lógica fue que al tratarse de una VPLS, la conectividad se haría a través de un mismo segmento de red, sin necesitar del ruteo por default realizado en el caso de los servicios IP. Ambas diferencias pueden encontrarse en las figuras 133 y 134, respectivamente, en la sección de diseño experimental. Al realizar la conectividad de esta forma e implementar la sonda RPM, puede verse en las figuras 181, 182 y 183 los resultados satisfactorios. En la Figura 184 puede observarse el reconocimiento de la dirección MAC del equipo de monitoreo dentro de la tabla VPLS. Esto ocurre debido a que se levantó una conexión IP desde dicho equipo. Lo mostrado en la figura comprueba cómo una VPLS realiza el reconocimiento de direcciones MAC. Una última diferencia es que en el caso de monitoreo IP, la VLAN de interconexión (entre equipo de monitoreo y PE) podía ser igual o no a la VLAN de conexión con la sede central. En este caso de monitoreo Ethernet, fue necesario que la VLAN de interconexión fuera la misma que la de conexión a la sede central. Esto tiene sentido, considerando que una VPLS emula una conexión Ethernet, todos los hosts deben pertenecer al mismo segmento de red para que exista la conectividad IP. Se realizaron pruebas colocando una VLAN distinta y la conectividad no fue exitosa. Como se discute más adelante, un monitoreo a nivel

IP siempre va a resultar más efectivo que un monitoreo a nivel MAC. La razón que impidió la implementación del monitoreo IP fue la clara dependencia que se tiene del suscriptor al tratarse de un servicio VPLS, un servicio que es transparente y en donde el proveedor no realiza conectividad IP ni utiliza protocolos de ruteo. Es necesario tener comunicación con el suscriptor, con el fin de que autorice el monitoreo, realice la configuración de una red (que sea como mínimo /29) para este fin, provea el direccionamiento y las direcciones IP destino y fuente para el proveedor. Con estas acciones, entonces el proveedor puede implementar el RPM utilizando la IP fuente. El primer inconveniente que se ve con esto es que resulta entonces necesario hablar con cada suscriptor para cada servicio, corriendo el riesgo de no recibir apoyo por parte del mismo, imposibilitando la aplicación del monitoreo. En ese sentido, de entrada se sabe que el proyecto no es escalable y probablemente no realizable para cada servicio. En segundo lugar, la dependencia que se tiene con el suscriptor, esperando que realice la configuración necesaria, indique el direccionamiento, la red a utilizar, y la distribución de IPs. En ese sentido se corre el riesgo de perder el monitoreo o de incluso pensar en la posibilidad de un incidente, si el mismo desconfigura la red. Por lo tanto, RPM tampoco resulta eficaz en este caso. En conclusión, se prefirió un monitoreo independiente del suscriptor y por ello se utilizó el sistema descrito en este trabajo. El uso de RPM se utilizó en VPLS sólo para casos críticos en los que el suscriptor fácilemente colaboró, ya que se sabe de la efectividad de este método.

Por último, se probó la eficacia del sistema de monitoreo MPLS VPLS en su totalidad en escenarios reales durante tres semanas, implementando el algoritmo y verificando el tráfico de los servicios al ser detectados fuera de operación. Las pruebas en escenarios reales se ejecutaron por entre 5 y 7 horas diarias. En el diseño experimental, puede observarse a detalle en qué consistieron las fases de esta etapa. Además de los mencionados PE centrales, designados en esta etapa como NAP1 y NAP2, se escogió otro PE, ER, para ser utilizado en esta etapa, incrementando el número de monitoreo de servicios. Fueron 118 servicios VPLS activos, luego de realizar la depuración. A pesar que en un inicio se estimó realizar el monitoreo para 50 servicios, se consideró que mientras más servicios se monitorearan, más era la posiblidad de detectar fallas y más sería la posibilidad de probar el sistema de monitoreo MPLS VPLS en escenarios reales. La depuración de servicios VPLS, por su parte, consitió en descartar los que no tuvieran conexión VPLS establecida, tabla MAC, o que la documentación sugiriera como servicio cancelado o aún no activado. El Cuadro 9 muestra un resumen de esto. Durante la etapa de pruebas se corrió el programa de detección y diagnóstico cada 10 minutos, generando ventanas emergentes sobre servicios VPLS que generaran cualquier tipo de alarmas planteadas. A pesar que se comprobó que el tiempo de vida de direcciones MAC en los equipos era el teórico por defecto (5 minutos), las pruebas realizadas a diferentes servicios VPLS (incluyendo el de pruebas), sugieren que una dirección MAC se pierde en un tiempo variable pero no máximo a 10 minutos, tal y como puede verse en el Cuadro 10. El resumen de escenarios de pruebas de tiempo de vida de direcciones MAC planteados se puede ver en el diseño experimental. Se probaron dos escenarios distintos en dos servicios VPLS en producción con RPM, deshabilitando la configuración de RPM con el fin de ver el comportamiento de pérdida y recuperación de las direcciones MAC implicadas en estos escenarios. Fue de mayor valor las pruebas realizadas en la VPLS de pruebas, ya que se pudo plantear una mayor cantidad de escenarios y permitió concluir que 10 minutos era un valor aceptable. Por otro lado, en el diseño experimental se observa a detalle los pasos realizados para la configuración de la VPLS de pruebas y sus puntos de conectividad IP. En el diagrama de la Figura 138 se resume lo realizado. Se representó la sesión iBGP entre los equipos NAP1 y NAP2 como la red MPLS en escenarios, reales, los equipos NAP1 y NAP2, como los sitios remotos de la VPLS, los switches, (en donde se lograron implementar, debido a la limitante en conexiones físicas) la red local de la operación remota del ISP, y los equipos de monitoreo, las sedes

del suscriptor, en donde se levanta conectividad IP y se genera tráfico (en este caso el tráfico se simuló implementando RPM hacia todas las “sedes”). El fin de esta configuración de VPLS de pruebas fue realizar pruebas técnicas y de tiempo de vida de direcciones MAC libremente, ya que está claro que con un servicio en producción las acciones a realizar son limitadas. Para las pruebas técnicas se probaron los escenarios planteados dos veces, una vez para cada equipo, NAP1 y NAP2, esperando resultados específicos en cada uno de ellos, como se ve planteado en el diseño experimental. En la sección de resultados, puede verse que se cumplió con lo esperado, concluyendo que el programa de detección y diagnóstico era técnicamente funcional.

El funcionamiento en escenarios reales del sistema de monitoreo MPLS VPLS fue igualmente importante que el técnico. En la sección de resultados se puede apreciar que el sistema de monitoreo es funcional, al detectar fallas en servicos VPLS en escenarios reales, con una efectividad global del 91.48% (Cuadro 1). En dicha sección, pueden verse los resultados por semana. Como se ve en la Cuadro 1, de los 35 casos presentados en las tres semanas de pruebas, 32 fueron exitosos (detectados con anticipación). Dos de ellos fueron no detectados. En ambos casos ocurrió con el mismo servicio, el cual, tiene dos direcciones MAC remotas, una del suscriptor y otra de la red local del ISP. Entonces el servicio al perder la MAC del suscriptor queda fuera de operación para el mismo, a pesar de ver aún una MAC remota (la del ISP en su operación local). Entonces ocurrió en ambos casos que previo a correr las pruebas, el servicio ya estaba fuera de operación, y el programa de detección y diagnóstico encontró aún una dirección MAC remota para el servicio, almacenándola en la base de datos “en memoria” y considerando el servicio operativo. Se concluye, para evitar estos casos, que al momento de implementar el programa de detección y diagnóstico, el mismo se compare con una base de datos como tal, la cual contendría todas las direcciones MAC de todos los servicios VPLS activos, al realizar un barrido en donde los mismos se encuentren operativos (al detectar direcciones MAC por cada sitio y con tráfico). Esto hubiese evitado no detectar los dos casos mencionados. El tercer caso, como se puede ver en el Cuadro 1, fue detectado pero sin anticipación. La explicación se basa en que se trata de un servicio VPLS crítico y por lo tanto con monitoreo IP por RPM, a cuyas gráficas, el suscriptor tiene acceso. El tiempo de detección del suscriptor fue de 5 minutos, mientras que el del ISP con el monitoreo de VPLS fue de aproximadamente 10 minutos. Como se ha mencionado antes, el monitoreo RPM es más eficiente. Por otro lado, puede verse que el tiempo de detección del sistema de monitoreo promedia 3.6 minutos. Esto debido a que hubo un servicio VPLS que permaneció fuera de operación durante las tres semanas y por tanto el tiempo de detección era inmediato cada día (cero minutos). Es importante resaltar en este punto que para servicios VPLS que quedaban fuera de operación durante la ejecución de las pruebas, el tiempo de detección varió entre 7 y 15 minutos, típicamente. Esto se puede observar en las figuras 185 y 186 en donde se muestra el tráfico y el ticket generado por un caso real de un servicio VPLS detectado fuera de operación. Se observa que la caída real (por tráfico) ocurrió a las 15:40GMT-6 y la apertura del ticket (que va de la mano con la detección del sistema), a las 15:55GMT-6. Por otra parte, en el Cuadro 2 se puede ver el resumen de alarmas generadas y su efectividad. Se puede ver que en el caso de alarmas rojas, las mismas son siempre efectivas. No sucede así con el resto y se ve particularmente que el número de alarmas naranjas es alto. Todas las alarmas naranjas, verdes y amarillas en falso, indican que los servicios no presentaron baja de tráfico y por tanto no presentaron una caída como tal. Se observó que cuando un servicio queda fuera de operación, sin importar su capacidad, el tráfico cae por debajo de los 100 bps, tanto de entrada como de salida. Hubo un único caso máximo que presento 2000bps de entrada, pero bien, 0bps de salida. El caso típico fue el de 30bps de entrada y 0bps de salida (ver Figura 186). Sin embargo otros casos comunes fueron los de 30bps de entrada y 30bps de salida, o bien 0bps de entrada y 30bps de salida. El caso mínimo fue de 0bps de entrada y 0bps de salida. La conclusión

fue entonces, definir un umbral para correr la aplicación del programa de detección y diagnóstico, para servicios VPLS activos cuyo tráfico de entrada o de salida disminuya de 100bps. Así en cualquiera de los casos mencionados (el máximo, el mínimo, el típico o los comunes), el programa se correría. Al tener entonces como primera herramienta el tráfico de los servicios VPLS y con el umbral definido, las alarmas verdes y amarillas prácticamente no se generarían (quedando en cero), y la mayor cantidad de alarmas naranjas tampoco. Esto en conjunto con la base de datos real de direcciones MAC para el registro histórico de comparación mencionada con anterioridad (llenada con la recolección de direcciones MAC de servicios VPLS detectados como operativos), las alarmas naranjas se reducirían a las de escenarios reales con falla. En el caso del Cuadro 2, pasarían de generarse 145 a únicamente 4 efectivas. Es importante notar que sin el registro histórico adecuado de comparación, no se generarían alarmas naranjas y no se detectarían las fallas de este tipo, tal y como sucedió en estas pruebas de tres semanas. En conclusión, el sistema de monitoreo MPLS VPLS debe estar conformado por tres partes fundamentales: la verificación del tráfico y el umbral definido, el programa de detección (y recolección) de direcciones MAC y diagnóstico, y la base de datos como tal (en sustitución de la base de datos "en memoria" implementada durante las pruebas). Con base en lo discutido, esto elevaría la efectividad del sistema de monitoreo VPLS a un 100% (sin contar los casos de servicios VPLS con monitoreo RPM implementado). Sin embargo, con los recursos utilizados durante las tres semanas de pruebas, el hecho de haberse anticipado en 32 de 35 ocasiones a los reportes de falla de suscriptores y tomar acciones (como se especificó en el diseño experimental) para solventar y así reducir tiempos de detección, diagnóstico y resolución, convierte a este sistema de monitoreo VPLS en un aporte de valor a la empresa, lo que mejoraría su imagen de cara a sus suscriptores.

El sistema de monitoreo para servicios MPLS IP-VPN y MPLS VPLS fue claramente dividido en dos subsistemas. La naturaleza de cada tipo de servicio y la necesidad de no depender del suscriptor obligó a realizarlo de esa forma. Un monitoreo basado en conectividad IP siempre será más eficiente que uno basado en direcciones MAC, debido a que se logran mejores resultados detectando eventos inmediatamente, con recursos más simples. Por ejemplo, el hecho de que es muy fácil segmentar una red grande por medio de la configuración de IPs, al configurar una sonda RPM con pruebas a lo largo de todo el camino que recorre para llegar a su destino, y por lo tanto, que no sólo sea inmediata la detección de la falla, sino que su ubicación exacta. Esto es practicamente imposible con monitoreo de capa 2 (en este Trabajo de Graduación, el objetivo del monitoreo de capa 2 fue únicamente la detección de la falla sobre los servicios). En IP, el complemento de generación de logs y gráficos, hace que se hable de un monitoreo completo y de nivel. Un monitoreo basado en MAC, como el mostrado en este trabajo es claramente menos eficiente, pero resulta eficaz y escalable (para capa 2) al no depender del suscriptor del servicio, como se discutió con anterioridad. Cada uno de los puntos discutidos a lo largo de este análisis para ambos subsistemas, refleja el cumplimiento de los objetivos específicos propuestos, colaborando con el objetivo general de este Trabajo de Graduación. A lo largo de la realización del trabajo, se percibió una mejora en el análisis y tiempo de diagnóstico de los servicios, debido a las herramientas implementadas (gráficas, syslog, diagnósticos automatizados) así como en la alerta del operador (gracias al portal de servicios implementado en parte con lo desarrollado en este Trabajo de Graduación y también gracias a las gráficas colocadas en las pantallas de monitoreo). Finalmente, se concluye que el sistema de monitoreo implementado contribuye a la reducción de los tiempos de respuesta y de diagnóstico de incidentes, transmitiendo al suscriptor una mejora en la calidad del soporte e imagen de la empresa.

IX. CONCLUSIONES

- Se cumplió con el objetivo general de este Trabajo de Graduación, basado en la realización de los objetivos específicos presentados.

- La naturaleza contrastante entre los enlaces MPLS IP-VPN y los MPLS-VPLS llevo a dividir el sistema de monitoreo en dos partes independientes.

- En servicios IP, el monitoreo basado en RPM resulta eficiente, eficaz y escalable.

- En servicios Ethernet, el monitoreo basado en RPM resulta eficiente, pero no eficaz ni escalable.

- En servicios Ethernet, se optó por el uso del monitoreo basado en direcciones MAC por encima del monitoreo basado en RPM.

- El monitoreo de servicios Ethernet e IP fue empleado por equipos fuera de la red MPLS.

- La base del monitoreo IP fue la aplicación del servicio RPM.

- Los pilares del monitoreo Ethernet fueron la generación de gráficas de tráfico, el algoritmo de diagnóstico inicial basado en la detección de direcciones MAC, y la base datos de las direcciones MAC de los servicios.

- Durante la etapa de pruebas del sistema de monitoreo Ethernet, la efectividad (detección con anticipación al suscriptor) fue de 91.43% y el tiempo de detección de 3.6 minutos.

- La aplicación adecuada de los tres pilares de monitoreo Ethernet mencionados elevaría la efectividad del sistema de monitoreo VPLS al 100% (descartando VPLS con RPM) y reduciría en su totalidad las alarmas en falos.

- El umbral de tráfico se definió para servicios VPLS que presenten una baja de 100 bps de entrada o de salida.

- El tiempo de vida teórico de direcciones MAC fue de 5 minutos y el práctico fue variable pero no máximo de 10 minutos.

- La configuración de una VPLS de pruebas contribuyó a obtener conclusiones precisas sobre el comportamiento de las direcciones MAC de una VPLS y permitió concluir que el programa de diagnóstico y detección fue totalmente confiable a nivel técnico.

- El sistema de monitoreo en general fue realizado para enlaces punto-punto y punto-multipunto.

- En el monitoreo de servicios IP, la conectividad lógica entre el equipo de monitoreo y la red MPLS se hizo únicamente a través del PE central.

- En el monitoreo de servicios Ethernet, la recolección de información fue independiente del PE seleccionado.

- En el monitoreo de servicios IP, la investigación de cada enlace resultó clave en asegurar una conectividad de punto a punto para cada uno.

- En el monitoreo de servicios IP, la VLAN seleccionada en el ruteo de RPM fue independiente de la utilizada entre el PE central y la sede central del suscriptor.

- La generación de gráficas RPM y de tráfico cumplió con agilizar el tiempo de respuesta del operador a incidentes.

- La utilización de Syslog para monitoreo IP mejoró la calidad del diagnóstico de incidentes temporales e históricos.

- La identificación de VRFs, VPLS y subinterfaces centrales ayudaron a disminuir el tiempo de diagnóstico.

- La nomenclatura implementada en el monitoreo de servicios IP contribuyo principalmente a la recolección y organización de información por medio del portal de servicios, y secundariamente a disminuir el tiempo de diagnóstico de incidentes.

- La gestión de equipos PE y de redes locales fue el método más efectivo en la investigación de servicios IP (cuando estuviera disponible, según el caso).

- En monitoreo de servicios IP, la conectividad vía ping fue obligatoria para la configuración de sondas RPM.

- El uso de estándares de parámetros para el monitoreo RPM obedeció a la criticidad de los servicios, diferenciados primordialmente por frecuencia de sondeo.

- La frecuencia del poleo de información para la generación de gráficas RPM y de tráfico representa una desventaja en el proceso de diagnóstico de históricos.

- En el sentido de alerta al operador, la herramienta de gráficas fue preferible con respecto a syslog.

- Los logs generados por pérdidas de ping individuales y de pruebas RPM totales, con propósitos distintos, permitieron en conjunto elaborar un diagnóstico de nivel.

- El estado de las conexiones de los sitios de una VPLS no es una alternativa funcional para el monitoreo de los servicios asociados.

- El reconocimiento de direcciones MAC por parte de un sitio VPLS ocurre cuando se levantan conexiones IP a través de las interfaces asociadas a esas direcciones MAC.

- El objetivo del algoritmo fue implementar un diagnóstico inicial y no uno detallado.

- La existencia de direcciones MAC provenientes de cada una de las interfaces lógicas activas no garantiza necesariamente la operatividad de todas las sedes remotas en una VPLS.

- El hecho de recibir una interfaz lógica remota (LSI) por sitio PE representó el punto débil de la implementación del algoritmo.

- El registro histórico buscó aliviar el punto débil de algoritmo no siendo efectivo en el cien por ciento de escenarios.

- La generación de gráficas de tráfico fue una de las bases del sistema de monitoreo VPLS y no puede ser considerada como un complemento.

- La evaluación del diagnóstico automatizado basado en el algoritmo desarrollado depende del consumo de los servicios VPLS.

- Se comprobó que SNMP no fue una herramienta funcional para el reconocimiento de direcciones MAC de VPLS.

- En la implementación del algoritmo para monitoreo Ethernet, fue clave identificar todas las interfaces lógicas activas antes que el desarrollo del diagnóstico como tal.

- Dentro del estado de las conexiones de una VPLS, las interfaces lógicas remotas recibidas (LSI) son una por PE o sitio remoto.

- Dentro del estado de las conexiones de una VPLS, no existe diferenciación entre las interfaces lógicas locales activas y de respaldo.

- Dentro de la configuración del sitio local, sí existe diferenciación entre las interfaces lógicas locales activas y de respaldo.

- El formato XML de Junos OS resultó preferible para la obtención de información sobre el formato ASCII.

- El formato XML presentado por Junos OS fue utilizado para obtener y realizar el parseo de la información de comandos de conexiones y tablas MAC de la VPLS.

- El formato ASCII de Junos OS fue utilizado para obtener y parsear información de comandos de configuración de VPLS.

- La primera parte del diagnóstico automatizado fue utilizada para determinar la inoperatividad de un servicio VPLS.

- La segunda parte del diagnóstico automatizado no garantiza necesariamente la operatividad o no de un servicio VPLS, pero provee advertencias de revisión avanzadas.

- La dependencia hacia el suscriptor descarta la posibilidad de aplicar un monitoreo RPM a todos los servicios VPLS.

- En un monitoreo RPM para servicios VPLS, la VLAN utilizada en el ruteo RPM fue dependiente de la VLAN utilizada según el servicio.

- Los puntos tratados a lo largo del trabajo reflejan el cumplimiento de los objetivos específicos.

- Sin importar los medios, el fin principal de una empresa de telecomunicaciones es transmitir una mejor calidad de soporte e imagen.

X. BIBLIOGRAFÍA

A. FUENTES IMPRESAS

Berkowitz, Howard. 2001. *WAN survival guide: strategies for VPNs and multiservice networks*. Nueva York: Wiley. 436 págs

González Morales, Alexandro. 2006. «Redes privadas virtuales». Trabajo de Graduación Universidad Autónoma del estado de Hidalgo. 186 págs

Arriola Díaz, Álvaro Xavier. 2013. «Implementación de Gestión Proactiva End-to-End para Servicios IP». Trabajo de Graduación Universidad del Valle de Guatemala. 77 págs

B. FUENTES DIGITALES

Alcatel Lucent. 2009. *VPLS Technical Tutorial*. Technology Whitepaper. [Julio, 2015] http://www.fplfibernet.com/pdf/VPLSTutorial.pdf

Beal, Vangie. 2015. *Circuit switching.* [Agosto, 2015] http://www.webopedia.com/TERM/C/circuit_switching.html

Beal, Vangie. 2015. *IP Internet Protocol.* [Agosto, 2015] http://www.webopedia.com/TERM/I/IP.html

Beal, Vangie. 2015. *Leased Line.* [Agosto, 2015] http://www.webopedia.com/TERM/L/leased_line.html

Beal, Vangie. 2015. *Packet Switching*. [Agosto, 2015] http://www.webopedia.com/TERM/P/packet_switching.html

BGP Expert. *What's BGP, Anyway?* [Agosto, 2015] http://www.bgpexpert.com/what.php

Brocade Communications System. 2009. *Technical Brief: Offering Scalable Layer 2 Services with VPLS and VLL.* [Agosto, 2015] http://www.brocadechina.com/download/MLX/Offering_Scalable_Layer2_Services_with_VPLS_and_VLL.pdf

Carroll, J. D. y J. Doyle. 2013. *BGP Message Formats*. [Agosto, 2015] http://flylib.com/books/en/2.295.1.29/1/

Cisco Systems. 2004. *Cisco IOS® MPLS Virtual Private LAN Service Business.* [Agosto, 2015] https://www.cisco.com/application/pdf/en/us/guest/tech/tk891/c1482/ccmigration_09186a00801ed507.pdf

Cisco Systems. 2004. *Cisco IOS® MPLS Virtual Private LAN Service (VPLS) Technical Deployment Overview.* [Agosto, 2015] https://www.cisco.com/application/pdf/en/us/guest/tech/tk891/c1482/ccmigration_09186a00801ed3ea.pdf

Cisco Systems. 2015. *Virtual Private LAN Services (VPLS).* [Agosto, 2015] http://www.cisco.com/c/en/us/td/docs/switches/lan/catalyst6500/ios/15-0SY/configuration/guide/15_0_sy_swcg/vpls.pdf

Juniper Networks. 2015. *Configuring RPM Probes*. [Junio, 2015]

http://www.juniper.net/documentation/en_US/junos12.2/topics/task/configuration/rpm-probes-configuring.html

Juniper Networks. 2010. *Configuring RPM Probes for BGP Monitoring*. [Junio, 2015] https://www.juniper.net/techpubs/software/junos-es/junos-es93/junos-es-admin-guide/configuring-rpm-probes-for-bgp-monitoring.html

Juniper Networks. 2015. *Junos PyEZ*. [Julio, 2015] https://techwiki.juniper.net/Automation_Scripting/010_Getting_Started_and_Reference/Junos_PyEZ

Juniper Networks. 2013. *Junos OS VPLS Configuration Guide*. [Julio, 2015] http://www.juniper.net/techpubs/en_US/junos13.1/information-products/pathway-pages/config-guide-vpns/config-guide-vpns-vpls.pdf

Juniper Networks. 2014. *Junos OS Configuration Overview*. [Junio, 2015] http://www.juniper.net/techpubs/en_US/junos14.2/topics/concept/syslog-messages-configuration-overview.html

Juniper Networks. 2014. *Junos OS VPLS Feature Guide for Routing Devices.* [Agosto, 2015] http://www.juniper.net/techpubs/en_US/junos14.2/information-products/pathway-pages/config-guide-vpns/config-guide-vpns-vpls.pdf

Juniper Networks. 2015. NETCONF XML Management Protocol Developer Guide. [Julio, 2015] http://www.juniper.net/techpubs/en_US/junos13.2/information-products/pathway-pages/netconf-guide/netconf.html#overview

Juniper Networks. 2015. RPM Overview. [Junio, 2015] http://www.juniper.net/documentation/en_US/junos12.1/topics/concept/security-rpm-overview.html

Juniper Networks. 2014. *Show VPLS Connections*. [Julio, 2015] http://www.juniper.net/documentation/en_US/junos12.1/topics/reference/command-summary/show-vpls-connections.html

Juniper Networks. 2013. *Show VPLS Mac-Table*. [Julio, 2015] https://www.juniper.net/documentation/en_US/junos13.1/topics/reference/command-summary/show-vpls-mac-table.html

Juniper Networks. 2015. *Test-interval*. [Junio, 2015] http://www.jnpr.net/techpubs/en_US/junos15.1/topics/reference/configuration-statement/test-interval-edit-services.html

Juniper Networks. 2015. *Thresholds*. [Junio, 2015] http://www.juniper.net/documentation/en_US/junos12.1x46/topics/reference/configuration-statement/thresholds-edit-services.html

Microsoft. 2015. *TCP/IP Protocol Architecture.* [Agosto, 2015] https://technet.microsoft.com/en-us/library/cc958821.aspx?f=255&MSPPError=-2147217396

Netgear. 2005. *Virtual Private Networking Basics*. [Julio, 2015] http://documentation.netgear.com/reference/nld/vpn/pdfs/FullManual.pdf

Networkers. 2015. *Junos and Python – Junos PyEZ – Part 1*. [Julio, 2015] http://www.networkers.fi/blog/junos-and-python-junos-pyez-part-1/

One Stop Click. *MPLS IP VNP Explained*. [Septiembre, 2015] http://networking.onestopclick.com/article/61/327/mpls-ip-vpn-explained.html

Pepelnjak, Ivan. *The Protocol that Makes the Internet Work*. [Agosto, 2015] http://searchtelecom.techtarget.com/feature/BGP-essentials-The-protocol-that-makes-the-Internet-work

Perkin, Roger. 2013. *Route Distinguisher vs Route Target – MPLS Tutorial*. [Septiembre, 2015]

http://www.rogerperkin.co.uk/ccie/mpls/route-distinguisher-vs-route-target/
RCF Editor. 2013. *Official Internet Protocol Standards*. [Agosto, 2015]
http://www.rfc-editor.org/rfc.html
Romero Ternero, Maricarmen. 2004. *Seguridad en redes y protocolos asociados*. [Agosto, 2015]
http://www.dte.us.es/personal/mcromero/docs/ip/tema-seguridad-IP.pdf
Ruiz González, José Luis. 2002. VPN - *Redes Privadas Virtuales*. [Agosto, 2015]
http://isa.uniovi.es/~sirgo/doctorado/VPN.pdf
Steenbergen, Richard A. *MPLS for Dummies*. [Agosto, 2015]
https://www.nanog.org/meetings/nanog49/presentations/Sunday/mpls-nanog49.pdf
Till Johnson, Johna. 2007. *MPLS Explained*. [Agosto, 2015]
http://www.networkworld.com/article/2297171/network-security/mpls-explained.html
ZTE. *Technical Whitepaper on MPLS L3VPN*. [Agosto, 2015]
http://wwwen.zte.com.cn/en/products/bearer/201308/P020130828527155850511.pdf

XI. ANEXOS

A. CÓDIGO DE PROGRAMA DE DESARROLLO DE ALGORITMO PARA VPLS

```
from jnpr.junos import Device
from lxml import etree
import sys
#Definicion DEVICE
dev = Device(host='IP', user='user', password='password', port=22)
#Login a DEVICE
try:
 dev.open()
except Exception as err:
 print err
 sys.exit(1)

        '''*****************************************************************************************************
                                                DEFINICION DE METODOS
*****************************************************************************************************'''

'''*****************************************************************************************************
1. get_vpls_facts
    ESTE METODO OBTIENE TODA LA INFORMACION NECESARIA Y DIAGNOSTICA EL ESTADO DE UNA VPLS.'''
'''***************************************************************************************************** '''
def get_vpls_facts():
'''*****************************************************************************************************  Definicion de
variables globales permanentes, 4 para cada servicio.'''
  global vpls_1, mac_1, if_1, overall_1, vpls_2, mac_2, if_2, overall_2 #.....etc.,
  ''' Variables booleanas de diagnóstico y de comparacion con variables permanentes'''
  global service_down, interface_presence, mac_count_less,mac_count_more,mac_address_change
  global mac_count, mac_count_current, ifs_current, macs_current, vpls_info_list, mac_table_list, logical_if_list, vpls_overall_message
  global vpls_update #Variable booleana que determina si la informacion de una VPLS debe actualizarse
'''*****************************************************************************************************
  *****************************************************************************************************
  VPLS get y count '''
  get_vpls_macs_rpc = dev.rpc.get_vpls_mac_table(logical_system ='Sistema') #Se obtienen todas las MAC Table disponibles
  get_vpls_macs_tostring = etree.tostring(get_vpls_macs_rpc) #El resultado del GET se convierte en string.
  cantidad_vpls_mx960 = get_vpls_macs_tostring.count('l2ald-mac-entry')/2  #El tag element ayuda a determinar la cantidad de VPLS Mac Table en el
equipo.
  vpls_service_mac_table_list = get_vpls_macs_tostring.split('l2ald-mac-entry', cantidad_vpls_mx960*int('2')) #Se separan los servicios VPLS del string que
los contiene todos separandolos en una lista
 '''*****************************************************************************************************

  *****************************************************************************************************
                                                VPLS Service: Parseo inicial
*****************************************************************************************************'''
  mac_count =vpls_selector = vpls_selector2=0
  while (vpls_selector < cantidad_vpls_mx960*int('2')): #Se garantiza que el ciclo dure en funcion de la cantidad de VPLS
    if (vpls_selector%2 != 0): #Se sabe que los strings deseados estan en los indices impares de la lista'''
      vpls_service_mac_table = vpls_service_mac_table_list[vpls_selector] # Se selecciona el string de la VPLS correspondiente'''
      mac_count = vpls_service_mac_table.count('l2-mac-address')/2 # Se determina la cantidad de MACs en la VPLS actual'''
      mac_count_string = str(mac_count)
      mac_count_current = mac_count    #Para evitar posibles errores en el programa.
      vpls_selector2 = vpls_selector2+1 #Enumera los servicios VPLS, solo incrementa en impares'''
   '''*****************************************************************************************************
                                    VPLS INFO: Se hacen listas iniciales parseando adecuadamente las variables deseadas
  *****************************************************************************************************'''
      vpls_name_parse_list = vpls_service_mac_table.split('l2-mac-routing-instance>', 2)
      logical_system_parse_list = vpls_service_mac_table.split('l2-mac-logical-system>',2)
      vlan_id_parse_list = vpls_service_mac_table.split('l2-bridge-vlan>',2)
      #Se obtiene el elemento deseado de la lista pero aun no esta completamente parseado
      vpls_name_parse_element = vpls_name_parse_list[1]
      logical_system_parse_element = logical_system_parse_list[1]
      vlan_id_parse_element = vlan_id_parse_list[1]
      #Se hace una lista para completar el parseo'''
      vpls_name_parse_list2 = vpls_name_parse_element.split('<', 1)
      logical_system_parse_list2 = logical_system_parse_element.split('<', 1)
      vlan_id_parse_list2 = vlan_id_parse_element.split('<', 1)
      #Se obtiene el elemento totalmente parseado como string'''
```

```
        vpls_name = vpls_name_parse_list2[0]
        logical_system = logical_system_parse_list2[0]
        vlan_id = vlan_id_parse_list2[0]
        #Lista con informacion general de la VPLS'''
        vpls_info_list = [vpls_name, logical_system, vlan_id, mac_count_string]
        '''*****************************************************************************************************
                                VPLS CONNECTIONS EXTENSIVE, determinar sitios remotos y sedes locales
    ***************************************************************************************************** '''
        #Se obtiene la informacion de VPLS connections de interes
        get_vpls_connection_rpc = dev.rpc.get_vpls_connection_information(logical_system ='Sistema', instance=vpls_name, extensive=True) #Se obtienen
todas las MAC Table disponibles
        vpls_service_connections = etree.tostring(get_vpls_connection_rpc)
        connect_count=0
        vpls_connections_list = vpls_service_connections.split('<label-block heading', 1)
        vpls_connections = vpls_connections_list[0]
        connect_count = vpls_connections.count('<interface-name>')
        #Lista de interfaces asociadas locales (para determinar cantidad de sedes) y remotas (para sitios remotos)
        interfaces_parse_list= vpls_connections.split('interface-name>', connect_count*2)
        count=count2=0
        interfaces_list= [None] * connect_count
        while (count < connect_count*2):
            if (count%2 != 0):
                interfaces_list0= interfaces_parse_list[count].split('<',1)
                interfaces_list[count2] = interfaces_list0[0]
                count2= count2+1
            count = count +1
        count = count2=0
        #Se determina la cantidad de interfaces locales y remotas (sitios totales=  1+interfaces remotas) (sedes locales = interfaces locales)
        connect_LSI_count=connect_Local_count=0
        while (count < connect_count):
            if(interfaces_list[count].startswith("lsi.") == True):
                connect_LSI_count=connect_LSI_count+1
            if(interfaces_list[count].startswith("ge-") == True or interfaces_list[count].startswith("xe-") == True):
                connect_Local_count=connect_Local_count+1
            count=count+1
        count=0
        '''VPLS Configuratiom  '''
        #Si hay mas de una interfaz local activa, se procede a revisar si esta configurada como primaria (1 sitio) o deben considerarse el resto
        #de interfaces como sedes individuales.
        if (connect_Local_count>1):
            configuration_vpls= dev.cli("show configuration logical-systems Sistema  routing-instances "+vpls_name)
            redundancia="active-interface"
            boolean= redundancia in configuration_vpls #Si existe el parametro es porque hay interfaz primaria, por ende una sola sede en el sitio local
            if(boolean==True):
                connect_Local_count=1
                interfaz_prim0=configuration_vpls.split('active-interface primary ',1)[1]
                interfaz_prim=interfaz_prim0.split(';',1)[0]
                while (count < connect_count):
                    if(interfaces_list[0].startswith("ge-") == True or interfaces_list[0].startswith("xe-") == True):
                        interfaces_list.pop(0)
                        connect_count=connect_count-1
                    count=count+1
                count=0
                interfaces_list.insert(0, interfaz_prim)
                connect_count=connect_count+1
        connect_count_string, connect_LSI_count_string, connect_Local_count_string = str(connect_count), str(connect_LSI_count), str(connect_Local_count)
        vplsconnections_info_list = [connect_count_string, connect_Local_count_string, connect_LSI_count_string]
        '''*****************************************************************************************************
                                VPLS MACS Y LOGICAL IFS: Se hacen listas iniciales de parseo, el mac_count es vital para este parseo
            ***************************************************************************************************** '''
        mac_table_parse_list= vpls_service_mac_table.split('l2-mac-address>', mac_count*2)
        logical_if_parse_list = vpls_service_mac_table.split('l2-mac-logical-interface>', mac_count*2)
        '''Ciclo de parseo y recoleccion de macs y logical ifs'''
        count=count2=0
        mac_table_list = [None]*mac_count
        logical_if_list= [None] * mac_count
        while (count < mac_count*2):
            if (count%2 != 0): #Si count es impar (residuo de division entre 2) ya que se sabe que las macs estan en los elementos impares de la lista'''
                mac_list0= mac_table_parse_list[count].split('<',1) #Se completa el parseo y se asigna a una lista temporal de dos elementos'''
                logical_list0 = logical_if_parse_list[count].split('<',1)
                mac_table_list[count2] = mac_list0[0] #Se asigna a la lista definitiva en la posicion count2'''
                logical_if_list[count2] = logical_list0[0]
                count2= count2+1 #Se incrementa el count solo cuando se obtienen las macs de la primer lista, es decir cuando count es impar'''
            count = count +1 #Este contador se incrementa cada ciclo'''
        count = count2=0
        ''''''***************************************************************************************************** '''
```

```
        '''*****************************************************************************************************
                                                VPLS OVERALL MESSAGE: Se despliega la informacion completa de la VPLS
                    ***************************************************************************************************** '''
        vpls_info_message = " "+str(vpls_selector2)+ ") "+vpls_info_list[0]+" overview: \n   Sistema logico: "+vpls_info_list[1]+ ", Vlan-id: "+vpls_info_list[2]+
"\n   Cantidad de MAC detectadas: "+vpls_info_list[3]+"\n"
        vpls_macs_message = "\n   VPLS Mac Table:  "+' | '.join(mac_table_list)
        vpls_ifs_message = "   VPLS Logical IF: "+'    |    '.join(logical_if_list) +"\n \n   Resultado del diagnostico: "
        connections_info_message = "   Sitios totales: "+str(connect_LSI_count+1)+ " (Locales=1, Remotos="+vplsconnections_info_list[2]+")"
        connections_sede= "   Interfaces locales: "+vplsconnections_info_list[1]
        connections_interfaces_message = "   Interfaces: "+', '.join(interfaces_list)
        vpls_overall_message =
"""+vpls_info_message+"\n"+connections_info_message+"\n"+connections_sede+"\n"+connections_interfaces_message+"\n"+vpls_macs_message+"\n"+v
pls_ifs_message+"\n"
        print "++++++++++++++++++++++++++++++++++++++++++++++++++"
        print vpls_overall_message
'''*****************************************************************************************************''
         '''*****************************************************************************************************
                                                PROCESO DE DIAGNOSTICO DE LA VPLS
                    *****************************************************************************************************'''
        vpls_update=False # Necesario que inicie en estado False para cada diagnostico de cada servicio VPLS '''
        '''1. VPLS Down por mac_count menor connect_count: Si hay menos macs que interfaces asociadas, el servicio se detecta de entrada caido'''
        if (mac_count<connect_count):
           service_down = True
           mac_count_less = True
        '''2. VPLS Down por LSI o Local: Si del paso 1, service_down=False, se revisa que provengan MACs de todas las subinterfaces asociadas a sitios.'''
        interfaces_boolean_lost_list = [None]*connect_count
        while(count<connect_count):
           while(count2<mac_count):
              if(interfaces_list[count]==logical_if_list[count2]):
                 interface_presence = True
                 count2=mac_count
              else:
                 interface_presence = False
                 count2=count2+1
           if(interface_presence==True):
              interfaces_boolean_lost_list[count]="False"
           if(interface_presence==False):
             interfaces_boolean_lost_list[count]="True"
           count2=0
           count=count+1
        count=0
        ''' 3. Resultado primer fase (pasos 1 y 2): Se determina si el servicio esta caido de acuerdo con informacion anterior '''
        while(count<connect_count):
           if (interfaces_boolean_lost_list[count]=="True"):
              service_down = True
              count=connect_count
           else:
              service_down = False
              count=count+1
        count=0
        ''' 4. Este metodo asigna las variables actuales a las variables permantentes correspondientes en funcion del nombre de la VPLS '''
        assign_permanent_variables()
        '''*****************************************************************************************************
                                                Mensajes de diagnostico
                    *****************************************************************************************************'''
        mac_count_less_than_interfaces_message = "      Este servicio se detecta caido. Existen menos direcciones MAC que sitios/interfaces asociadas a la
VPLS. Debe provenir al menos una MAC por sitio/interfaz. "# El numero de MACs default es: "+ str(mac_count_current)
        no_interfaces_message0 = "      Este servicio se detecta caido. Existen sitios/interfaces sin direcciones MAC asociadas. "
        no_interfaces_message = "      No se reciben direcciones MAC de las siguientes interfaces: "# El numero default de MACs es:
"+str(mac_count_current)
        vpls_up_conditional = "      Servicio probablemente operativo. Se detecta MACs de ambos puntos pero se tienen las siguientes observaciones: "
        mac_count_same_message = "       > El numero de MACs es el esperado: " + str(mac_count_current) + ", pero se detectan cambios de MACS."
        mac_count_more_message = "       > Existen mas direcciones MAC de las esperadas, el numero de MACs default es: "+str(mac_count_current)
        mac_count_less_message = "       > Existen menos direcciones MAC de las esperadas, el numero de MACs default es: "+str(mac_count_current)
        vpls_update_message = "       > Sin embargo, las direcciones MAC originales se mantienen por lo que se detecta servicio operativo y se procede a
actualizar la informacion de esta VPLS."
        vpls_ok_message = "      Servicio se detecta operativo. Se detecta propagacion de MACs de todos los puntos del servicio, las MACs son las minimas
esperadas y el MAC count ("+str(mac_count_current)+") tambien."
     '''*****************************************************************************************************'''
        mac_boolean_change_list = [None]*mac_count_current #Lista booleana para identificar las MACs originales que ya no se observan en la busqueda
actual.'''
        '''*****************************************************************************************************
             5. Evaluacion del MAC Count: Si se detecta servicio operativo, entonces compara el mac count actual con el original.
     *****************************************************************************************************'''
        if (service_down==False):
           if(mac_count_current==mac_count):
```

```
            mac_count_more, mac_count_less=False, False
        if(mac_count>mac_count_current):
            mac_count_more,mac_count_less=True,False
        if(mac_count<mac_count_current):
            mac_count_more,mac_count_less=False,True
        '''*************************************************************************************************************'''
        '''*************************************************************************************************************
            6. Evaluacion de Mac addresses: Si se detecta servicio operativo, entonces buscan las macs originales en el barrido actual de macs
 ************************************************************************************************************* '''
        count_origin= count_now =0
        while(count_origin<mac_count_current):
            while(count_now<mac_count):
                if (macs_current[count_origin]==mac_table_list[count_now]):
                    mac_address_change=False
                    count_now=mac_count
                else:
                    mac_address_change=True
                    count_now=count_now+1
            if(mac_address_change==True):
                mac_boolean_change_list[count_origin]="True"
            if(mac_address_change==False):
                mac_boolean_change_list[count_origin]="False"
            count_now=0
            count_origin=count_origin+1
        count_origin=0
'''*************************************************************************************************************
        *************************************************************************************************************
        7. Mensajes a desplegar de acuerdo a diagnostico
*************************************************************************************************************'''
        mac_address_change_message = "None"  #Se define asi para identificar si este cambio se da o no '''
        interfaces_lost_message ="None"
        '''7.1 Mensajes para servicio detectado como caido '''
        if (service_down==True):
            if (mac_count_less==True):
                print mac_count_less_than_interfaces_message
                print no_interfaces_message
                while(count_origin<connect_count):
                    if(interfaces_boolean_lost_list[count_origin]=="True"):
                        interfaces_lost_message = "       > Interfaz: "+interfaces_list[count_origin] +""
                        print interfaces_lost_message
                    count_origin = count_origin+1
                count_origin=0
            else:
                print no_interfaces_message0
                print no_interfaces_message
                while(count_origin<connect_count):
                    if(interfaces_boolean_lost_list[count_origin]=="True"):
                        interfaces_lost_message = "       > Interfaz: "+interfaces_list[count_origin] +""
                        print interfaces_lost_message
                    count_origin = count_origin+1
                count_origin=0
        ''' 7.2 Mensajes para servicio que se detecta operativo pero con cambios en mac addresses y mac count '''
        if (service_down == False):
            if (mac_count_less == True):
                print vpls_up_conditional + "\n"+mac_count_less_message
            if (mac_count_more == True):
                print vpls_up_conditional + "\n"+mac_count_more_message
                vpls_update= True  # Pero ademas de que hayan mas MACs tambien debe cumplirse que las originales esten presentes: '''
            while(count_origin<mac_count_current):
                if(mac_boolean_change_list[count_origin]=="True"):
                    if(mac_count_less==False and mac_count_more == False):
                        if(count==0):
                            print vpls_up_conditional + "\n"+mac_count_same_message  #Si hay cambio de MAC addresses, count se mantiene'''
                        count=count+1
                    vpls_update=False # Si falta una MAC la condicion para actualizar ya no se cumple'''
                    mac_address_change_message = "            > La direccion MAC: "+macs_current[count_origin]+ ", deberia provenir de la interfaz logica:
"+ifs_current[count_origin]
                    print mac_address_change_message
                count_origin = count_origin+1
            count_origin=count=0
            ''' Si hay mas macs y las originales se mantienen entonces actualiza la informacion de la VPLS en evaluacion '''
            if (vpls_update==True):
                print vpls_update_message
                assign_permanent_variables()
        ''' 7.3 Mensaje para servicio que se detecta operativo sin cambios de mac addresses ni mac count '''
        if (service_down == False and mac_count_more == False and mac_count_less == False and mac_address_change_message=="None"):
```

```
            print vpls_ok_message
          #print mac_boolean_change_list #print con fin unico de control en la evaluacion de este programa
          print "++++++++++++++++++++++++++++++++++++++++++++++++\n"
'''*************************************************************************************************************'''
      '''*************************************************************************************************************'''
      vpls_selector = vpls_selector+1 #Se procede a revisar el siguiente servicio'''
      time.sleep(1) '''************************************************************************************************************* '''
   return
'''*************************************************************************************************************'''

'''*************************************************************************************************************
2. assing_permanent_variables: ESTE METODO ALMACENA LA INFORMACION EN VARIABLES PERMANENTES UNA VEZ SI EL SERVICIO ESTA ARRIBA Y OTRA SI SE DEBE ACTUALIZAR LA INFORMACION.
*************************************************************************************************************'''
def assign_permanent_variables():
'''*************************************************************************************************************  Definicion de variables globales permanentes '''
  Definicion de variables globales permanentes, 4 para cada servicio.'''
   global vpls_1, mac_1, if_1, overall_1, vpls_2, mac_2, if_2, overall_2 #.....etc.,
   ''' Variables booleanas de diagnóstico y de comparacion con variables permanentes'''
   global service_down, lsi_presence, ge_presence, mac_count_less,mac_count_more,mac_address_change
   global mac_count, mac_count_current, ifs_current, macs_current, vpls_info_list, mac_table_list, logical_if_list, vpls_overall_message
   global vpls_update
*************************************************************************************************************'''
   if (vpls_info_list[0]=="VPLS1"): #1
      if((service_down==False and if_1==0) or vpls_update==True):
         vpls_1, mac_1, if_1, overall_1 = vpls_info_list, mac_table_list, logical_if_list, vpls_overall_message
         mac_count_current, ifs_current, macs_current = int(vpls_1[3]), if_1, mac_1
      elif(service_down==False and if_1!=0):
         mac_count_current, ifs_current, macs_current = int(vpls_1[3]), if_1, mac_1
   elif (vpls_info_list[0]=="VPLS2 "):#2
      if((service_down==False and if_2==0) or vpls_update==True):
         vpls2, mac_2, if_2, overall_2 = vpls_info_list, mac_table_list, logical_if_list, vpls_overall_message
         mac_count_current, ifs_current, macs_current = int(vpls_2 [3]), if_2, mac_2
      elif(service_down==False and if_2!=0):
         mac_count_current, ifs_current, macs_current = int(vpls_2[3]), if_2, mac_2
   elif (vpls_info_list[0]=="VPLS3"):#3……..ETC.
…
return
'''*************************************************************************************************************
                                                MAIN: MONITOREO DE SERVICIOS VPLS
*************************************************************************************************************'''
Titulo= "-------------------------------------------------------------------\n-                MONITOREO DE SERVICIOS VPLS                -\n-------------------------------------------------------------\n"
print Titulo
infinite_loop=1
'''*************************************************************************************************************
    DEFINICION DE VARIABLES GLOBALES
*************************************************************************************************************'''
'''PERMANENTES'''
vpls_1= mac_1= if_1= overall_1 = vpls_1= mac_1= if_1= overall_1= 0 #ET….
'''DE DIAGNOSTICO Y COMPARACION'''
service_down=interface_presence=mac_count_less=mac_count_more=mac_address_change=True
vpls_update=False
mac_count=mac_count_current= ifs_current= macs_current=vpls_info_list=mac_table_list=logical_if_list=vpls_overall_message=0
'''*************************************************************************************************************

*************************************************************************************************************
    CICLO INFINITO
*************************************************************************************************************'''
while (infinite_loop ==1):
   get_vpls_facts()
   print "-------------------------------------------------------------------\n"
   time.sleep(4)
'''*************************************************************************************************************'''
dev.close()
```

B. INSTALACIÓN DE PYEZ Y SUS DEPENDENCIAS

A continuación, se resumen los pasos para la instalación de las herramientas de desarrollo que permitieron la implementación del algoritmo de diagnóstico de servicios VPLS. A través de estas herramientas, se realiza la interacción con dispositivos Juniper dentro de un sistema operativo Windows.

1. Descargar e instalar Python 2.7

   a. Ir al sitio https://www.python.org/downloads/, y descargar la vesión 2.7.10 o similar. Esto descargará un instalador o paquete de formato MSI.

Figura#216: Descargar Python 2.7

   b. Doble click al paquete MSI descargado y seguir los pasos de instalación.

Figura#217 Instalar Python 2.7

2. Setup tools + PIP: el software de Setup tools permite y facilita instalar cualquier dependencia de Python mediante línea de comandos en complemento PIP. Ir al sitio: http://docs.python-guide.org/en/latest/starting/install/win/, descargar y correr los programas *ez_setup.py* y *get-pip.py*. Es recomendado guardar y correrlos desde la carpeta scripts, creada al momento de instalar Python en el directorio especificado.

Figura#218: Setup tools + PIP

Setuptools + Pip

The most crucial third-party Python software of all is Setuptools, which extends the packaging and installation facilities provided by the distutils in the standard library. Once you add Setuptools to your Python system you can download and install any compliant Python software product with a single command. It also enables you to add this network installation capability to your own Python software with very little work.

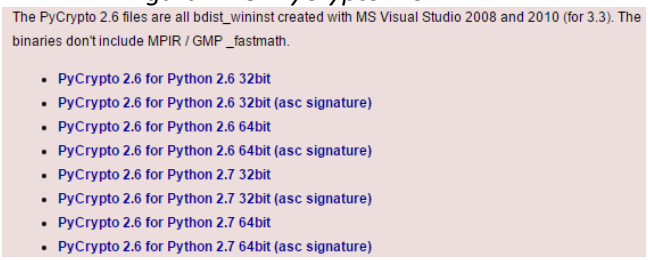
To obtain the latest version of Setuptools for Windows, run the Python script available here: ez_setup.py

You'll now have a new command available to you: **easy_install**. It is considered by many to be deprecated, so we will install its replacement: **pip**. Pip allows for uninstallation of packages, and is actively maintained, unlike easy_install.

To install pip, run the Python script available here: get-pip.py

3. PyCrypto 2.6: ir al sitio http://www.voidspace.org.uk/python/modules.shtml#pycrypto, descargar e instalar "*PyCrypto 2.6 for Python 2.7 64 bit.*"

Figura#219: PyCrypto 2.6

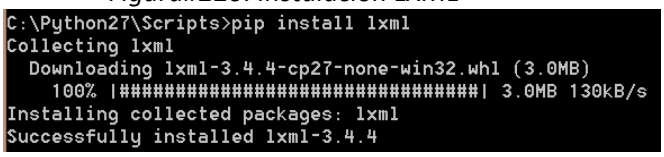


The PyCrypto 2.6 files are all bdist_wininst created with MS Visual Studio 2008 and 2010 (for 3.3). The binaries don't include MPIR / GMP _fastmath.

- PyCrypto 2.6 for Python 2.6 32bit
- PyCrypto 2.6 for Python 2.6 32bit (asc signature)
- PyCrypto 2.6 for Python 2.6 64bit
- PyCrypto 2.6 for Python 2.6 64bit (asc signature)
- PyCrypto 2.6 for Python 2.7 32bit
- PyCrypto 2.6 for Python 2.7 32bit (asc signature)
- PyCrypto 2.6 for Python 2.7 64bit
- PyCrypto 2.6 for Python 2.7 64bit (asc signature)

4. Instalación LXML : utilizar el comando: "*pip install lxml*" desde el directorio en donde se haya instalado Python, dentro de la carpeta "*Scritps*". Por ejemplo: *C:\Python27\scripts.*

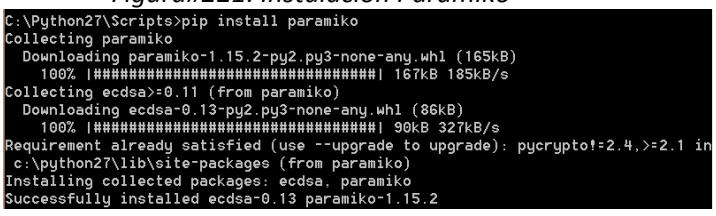
Figura#220: Instalación LXML



```
C:\Python27\Scripts>pip install lxml
Collecting lxml
  Downloading lxml-3.4.4-cp27-none-win32.whl (3.0MB)
    100% |################################| 3.0MB 130kB/s
Installing collected packages: lxml
Successfully installed lxml-3.4.4
```

5. Instalación Paramiko: similarmente, con el comando "*pip install paramiko*".

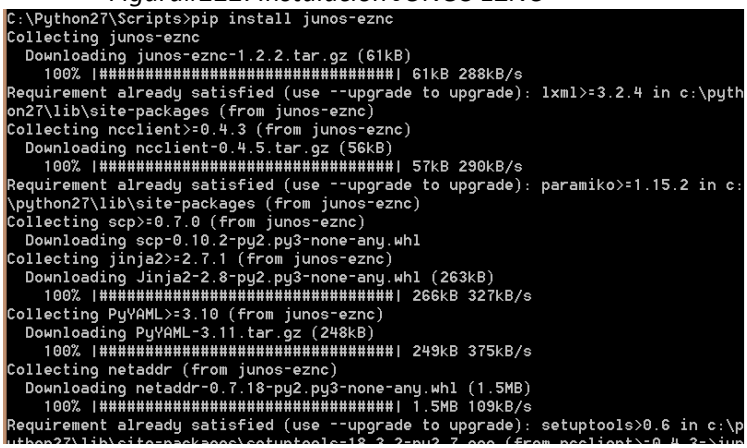
Figura#221: Instalación Paramiko



```
C:\Python27\Scripts>pip install paramiko
Collecting paramiko
  Downloading paramiko-1.15.2-py2.py3-none-any.whl (165kB)
    100% |################################| 167kB 185kB/s
Collecting ecdsa>=0.11 (from paramiko)
  Downloading ecdsa-0.13-py2.py3-none-any.whl (86kB)
    100% |################################| 90kB 327kB/s
Requirement already satisfied (use --upgrade to upgrade): pycrypto!=2.4,>=2.1 in
 c:\python27\lib\site-packages (from paramiko)
Installing collected packages: ecdsa, paramiko
Successfully installed ecdsa-0.13 paramiko-1.15.2
```

6. Instalación JUNOS EZNC: de forma similar, con el comando "*pip install junos-eznc*".

Figura#222: Instalación JUNOS EZNC



7. Pruebas del programa: A continuación puede verse la programación y correcto funcionamiento de un programa luego de la instalación.

Figura#223: Pruebas del programa (1)

```
from pprint import pprint
from jnpr.junos import Device
dev = Device(host='[redacted]', user='jrobles', password='[redacted]', port=22)
dev.open()
pprint(dev.facts)
dev.close()
```

Figura#224: Pruebas del programa (2)

```
Python 2.7.10 (default, May 23 2015, 09:40:32) [MSC v.1500 32 bit (Intel)] on wi
n32
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
{'2RE': True,
 'HOME': '/var/home/one_network',
 'RE0': {'last_reboot_reason': 'Router rebooted after a normal shutdown.',
         'mastership_state': 'master',
         'model': 'RE-S-1800x4',
         'status': 'OK',
         'up_time': '251 days, 17 hours, 39 minutes, 26 seconds'},
 'RE1': {'last_reboot_reason': 'Router rebooted after a normal shutdown.',
         'mastership_state': 'backup',
         'model': 'RE-S-1800x4',
         'status': 'OK',
         'up_time': '251 days, 18 hours, 25 seconds'},
 'domain': None,
 'fqdn': '[redacted]',
 'hostname': '[redacted]',
 'ifd_style': 'CLASSIC',
 'master': 'RE0',
 'model': 'MX960',
 'personality': 'MX',
 'serialnumber': 'JN11C8ABFAFA',
 'switch_style': 'BRIDGE_DOMAIN',
 'vc_capable': False,
 'version': '12.3R7.7',
 'version_RE0': '12.3R7.7',
 'version_RE1': '12.3R7.7',
 'version_info': junos.version_info(major=(12, 3), type=R, minor=7, build=7)}
>>>
```

C. CONFIGURACIÓN VPLS DE PRUEBAS

M10i#1:

```
{master}
jrobles@M10i#1> show configuration | match VPLS-MONITOREO-PRUEBA_TESIS | display set
set routing-instances VPLS-MONITOREO-PRUEBA_TESIS_RPM description "RPM temporal para pruebas de monitoreo de VPLS hacia NAP1"
set routing-instances VPLS-MONITOREO-PRUEBA_TESIS_RPM instance-type virtual-router
set routing-instances VPLS-MONITOREO-PRUEBA_TESIS_RPM interface fe-0/2/8.804
set routing-instances VPLS-MONITOREO-PRUEBA_TESIS_RPM2 description "RPM temporal para pruebas de monitoreo de VPLS hacia NAP1"
set routing-instances VPLS-MONITOREO-PRUEBA_TESIS_RPM2 instance-type virtual-router
set routing-instances VPLS-MONITOREO-PRUEBA_TESIS_RPM2 interface ge-1/3/0.804
set services rpm probe VPLS-MONITOREO-PRUEBA_TESIS_RPM test VPLS-MONITOREO_RPM-ping2 target address 10.10.10.5
set services rpm probe VPLS-MONITOREO-PRUEBA_TESIS_RPM test VPLS-MONITOREO_RPM-ping2 routing-instance VPLS-MONITOREO-PRUEBA_TESIS_RPM2
set services rpm probe VPLS-MONITOREO-PRUEBA_TESIS_RPM test VPLS-MONITOREO_RPM-ping target address 10.10.10.2
set services rpm probe VPLS-MONITOREO-PRUEBA_TESIS_RPM test VPLS-MONITOREO_RPM-ping routing-instance VPLS-MONITOREO-PRUEBA_TESIS_RPM

{master}
jrobles@M10i#1> show configuration interfaces fe-0/2/8.804
description "Interfaz para RPM temporal para pruebas de monitoreo de VPLS";
vlan-id 804;
family inet {
   address 10.10.10.1/29;
}

{master}
jrobles@ M10i#1> show configuration interfaces ge-1/3/0.804
description "Interfaz para RPM temporal para pruebas de monitoreo de VPLS hacia NAP1";
vlan-id 804;
family inet {
   address 10.10.10.3/29;
}
```

EX#1:

```
{master:0}
jrobles@EX#1> show configuration | match VPLS-MONITOREO-PRUEBA_TESIS | display set
set interfaces ge-0/0/47 unit 0 family ethernet-switching vlan members VPLS-MONITOREO-PRUEBA_TESIS
set interfaces ge-0/1/3 unit 0 family ethernet-switching vlan members VPLS-MONITOREO-PRUEBA_TESIS
set vlans VPLS-MONITOREO-PRUEBA_TESIS description "VLAN Temporal para pruebas de VPLS"
set vlans VPLS-MONITOREO-PRUEBA_TESIS vlan-id 804
```

NAP1:

```
@NAP1> show configuration | match VPLS-MONITOREO-PRUEBA_TESIS | display set
set logical-systems SISTEMA policy-options policy-statement POLITICA-exp term VPLS from community VPLS-MONITOREO-PRUEBA_TESIS
set logical-systems SISTEMA policy-options policy-statement POLITICA-imp term VPLS from community VPLS-MONITOREO-PRUEBA_TESIS
set logical-systems SISTEMA policy-options community VPLS-MONITOREO-PRUEBA_TESIS members target:26617:8060
set logical-systems SISTEMA routing-instances VPLS-MONITOREO-PRUEBA_TESIS description "VPLS temporal para pruebas de monitoreo"
set logical-systems SISTEMA routing-instances VPLS-MONITOREO-PRUEBA_TESIS instance-type vpls
set logical-systems SISTEMA routing-instances VPLS-MONITOREO-PRUEBA_TESIS vlan-id 804
```

set logical-systems SISTEMA routing-instances VPLS-MONITOREO-PRUEBA_TESIS interface ge-1/0/0.804
set logical-systems SISTEMA routing-instances VPLS-MONITOREO-PRUEBA_TESIS interface ge-1/0/1.804
set logical-systems SISTEMA routing-instances VPLS-MONITOREO-PRUEBA_TESIS interface ge-1/1/1.804
set logical-systems SISTEMA routing-instances VPLS-MONITOREO-PRUEBA_TESIS route-distinguisher x.x.134.1:8060
set logical-systems SISTEMA routing-instances VPLS-MONITOREO-PRUEBA_TESIS vrf-target target:xxxxx:8060
set logical-systems SISTEMA routing-instances VPLS-MONITOREO-PRUEBA_TESIS protocols vpls site-range 20
set logical-systems SISTEMA routing-instances VPLS-MONITOREO-PRUEBA_TESIS protocols vpls no-tunnel-services
set logical-systems SISTEMA routing-instances VPLS-MONITOREO-PRUEBA_TESIS protocols vpls site Prueba-1 site-identifier 1

@NAP1> show configuration logical-systems SISTEMA interfaces ge-1/0/0.804
description "VPLS temporal para pruebas de monitoreo";
encapsulation vlan-vpls;
vlan-id 804;
family vpls;
@NAP1> show configuration logical-systems SISTEMA interfaces ge-1/0/1.804
description "VPLS temporal para pruebas de monitoreo";
encapsulation vlan-vpls;
vlan-id 804;
family vpls;
@NAP1> show configuration logical-systems SISTEMA interfaces ge-1/1/1.804
description "VPLS temporal para pruebas de monitoreo";
encapsulation vlan-vpls;
vlan-id 804;
family vpls;

NAP2:
@NAP2> show configuration | match VPLS-MONITOREO-PRUEBA_TESIS | display set
set logical-systems SISTEMA policy-options policy-statement POLITICA-exp term VPLS-Prueba-Monitoreo from community VPLS-MONITOREO-PRUEBA_TESIS
set logical-systems SISTEMA policy-options policy-statement POLITICA-imp term VPLS-Prueba-Monitoreo from community VPLS-MONITOREO-PRUEBA_TESIS
set logical-systems SISTEMA policy-options community VPLS-MONITOREO-PRUEBA_TESIS members target:26617:8060
set logical-systems SISTEMA routing-instances VPLS-MONITOREO-PRUEBA_TESIS description "VPLS temporal para pruebas de monitoreo"
set logical-systems SISTEMA routing-instances VPLS-MONITOREO-PRUEBA_TESIS instance-type vpls
set logical-systems SISTEMA routing-instances VPLS-MONITOREO-PRUEBA_TESIS vlan-id 804
set logical-systems SISTEMA routing-instances VPLS-MONITOREO-PRUEBA_TESIS interface ge-1/0/1.804
set logical-systems SISTEMA routing-instances VPLS-MONITOREO-PRUEBA_TESIS interface ge-1/1/1.804
set logical-systems SISTEMA routing-instances VPLS-MONITOREO-PRUEBA_TESIS route-distinguisher x.x.134.2:8060
set logical-systems SISTEMA routing-instances VPLS-MONITOREO-PRUEBA_TESIS vrf-target target:xxxx:8060
set logical-systems SISTEMA routing-instances VPLS-MONITOREO-PRUEBA_TESIS protocols vpls site-range 20
set logical-systems SISTEMA routing-instances VPLS-MONITOREO-PRUEBA_TESIS protocols vpls no-tunnel-services
set logical-systems SISTEMA routing-instances VPLS-MONITOREO-PRUEBA_TESIS protocols vpls site Prueba-2 site-identifier 5

@NAP2> show configuration logical-systems SISTEMA interfaces ge-1/0/1.804
description "VPLS temporal para pruebas de monitoreo";
encapsulation vlan-vpls;
vlan-id 804;
family vpls;
@NAP2> show configuration logical-systems SISTEMA interfaces ge-1/1/1.804
description "VPLS temporal para pruebas de monitoreo";
encapsulation vlan-vpls;
vlan-id 804;
family vpls;

EX#2:
{master:0}

```
jrobles@EX#2> show configuration | match VPLS-MONITOREO-PRUEBA_TESIS | display set
set interfaces ge-0/0/47 unit 0 family ethernet-switching vlan members VPLS-MONITOREO-PRUEBA_TESIS
set interfaces ge-0/1/2 unit 0 family ethernet-switching vlan members VPLS-MONITOREO-PRUEBA_TESIS
set vlans VPLS-MONITOREO-PRUEBA_TESIS description "VLAN Temporal para pruebas de VPLS"
set vlans VPLS-MONITOREO-PRUEBA_TESIS vlan-id 804
```

M10i#4:

```
@M10i#4 > show configuration | match VPLS-MONITOREO-PRUEBA_TESIS | display set
set routing-instances VPLS-MONITOREO-PRUEBA_TESIS_RPM description "RPM temporal para pruebas de monitoreo de
VPLS hacia NAP2"
set routing-instances VPLS-MONITOREO-PRUEBA_TESIS_RPM instance-type virtual-router
set routing-instances VPLS-MONITOREO-PRUEBA_TESIS_RPM interface fe-1/1/1.804
set routing-instances VPLS-MONITOREO-PRUEBA_TESIS_RPM2 description "RPM temporal para pruebas de monitoreo
de VPLS hacia NAP1"
set routing-instances VPLS-MONITOREO-PRUEBA_TESIS_RPM2 instance-type virtual-router
set routing-instances VPLS-MONITOREO-PRUEBA_TESIS_RPM2 interface ge-0/3/3.804
set routing-instances VPLS-MONITOREO-PRUEBA_TESIS_RPM3 description "RPM temporal para pruebas de monitoreo
de VPLS hacia NAP2"
set routing-instances VPLS-MONITOREO-PRUEBA_TESIS_RPM3 instance-type virtual-router
set routing-instances VPLS-MONITOREO-PRUEBA_TESIS_RPM3 interface ge-1/3/3.804
set services rpm probe VPLS-MONITOREO-PRUEBA_TESIS_RPM test VPLS-MONITOREO_RPM-ping2 target address
10.10.10.3
set services rpm probe VPLS-MONITOREO-PRUEBA_TESIS_RPM test VPLS-MONITOREO_RPM-ping2 routing-instance
VPLS-MONITOREO-PRUEBA_TESIS_RPM2
set services rpm probe VPLS-MONITOREO-PRUEBA_TESIS_RPM test VPLS-MONITOREO_RPM-ping3 target address
10.10.10.4
set services rpm probe VPLS-MONITOREO-PRUEBA_TESIS_RPM test VPLS-MONITOREO_RPM-ping3 routing-instance
VPLS-MONITOREO-PRUEBA_TESIS_RPM3
set services rpm probe VPLS-MONITOREO-PRUEBA_TESIS_RPM test VPLS-MONITOREO_RPM-ping target address
10.10.10.1
set services rpm probe VPLS-MONITOREO-PRUEBA_TESIS_RPM test VPLS-MONITOREO_RPM-ping routing-instance VPLS-
MONITOREO-PRUEBA_TESIS_RPM

@M10i#4> show configuration interfaces fe-1/1/1.804
description "Interfaz para RPM temporal para pruebas de monitoreo de VPLS";
vlan-id 804;
family inet {
   address 10.10.10.2/29;
}

@M10i#4> show configuration interfaces ge-0/3/3.804
description "Interfaz para RPM temporal para pruebas de monitoreo de VPLS hacia MX960#1";
vlan-id 804;
family inet {
   address 10.10.10.4/29;
}

@M10i#4> show configuration interfaces ge-1/3/3.804
description "Interfaz para RPM temporal para pruebas de monitoreo de VPLS hacia MX960#2";
vlan-id 804;
family inet {
   address 10.10.10.5/29;
}
```

D. CRONOGRAMA DE ACTIVIDADES

2015		Junio				Julio					Agosto				Septiembre					Octubre	
Fase	Parte	Sem 1-7	Sem 8-14	Sem 15-21	Sem 22-28	Sem 29-5	Sem 6-12	Sem 13-19	Sem 20-26	Sem 27-2	Sem 3-9	Sem 10-16	Sem 17-23	Sem 24-30	Sem-31-6	Sem 6-12	Sem 13-19	Sem 20-26	Sem 27-3	Sem 4-10	Sem 11-17
Inicial	**Selección y delimitación del tema**																				
	Realización de protocolo de tesis																				
	Aprobación de protocolo																				
Fase IP-VPN	**Generar un listado de enlaces IP-VPN**																				
	Registro de enlaces IP-VPN monitoreados																				
	Aplicar RPMz en IP-VPN																				
	Creación de archivos Syslog																				
	Enlaces IP-VPN con problema de ping o OPE																				
	Identificación de VRFs y subinterfaces																				
	Guía estandarizada IP-VPN																				
Fase Ethernet	Fase investigativa de lectura de trafico																				
	Fase investigativa detección de MACs																				
	Implementación sobre enlaces Ethernet																				
	Registro automatizado de MACs																				
	Probar de funcionamiento y edificio																				
	Identificación de VRFs y subinterfaces																				
	Guía estandarizada Ethernet																				
Trabajo escrito	Indice																				
	Resumen del trabajo e introducción																				
	Justificación																				
	Objetivos generales y específicos																				
	Marco teórico																				
	Metodología																				
	Diseño Experimental																				
	Resultados																				
	Discusión																				
	Conclusiones y recomendaciones																				
	Otras partes del trabajo escrita																				
	Recopilación de fuentes de información																				
Final	Revisiones y correcciones de implementación																				
	Revisiones y correcciones trabajo escrita																				
	Aprobación trabajo práctico																				
	Aprobación trabajo escrita																				
	Establecer de fecha de presentación																				

XII. GLOSARIO

A

- API: *Application Program Interface,* conjunto de protocolos, rutinas y herramientas para construir aplicaciones de software.
- Asíncrono: en telecomunicaciones, se refiere a la transmisión de información sin la señal de un reloj externo.

B

- Bits: unidad básica de la información en comunicaciones digitales y denominación de datos en capa 1.
- Bucle: en switches, se refiere a un ciclo infinito en la distribución de tramas.
- Bytes: 8 bits.

C

- Calidad de servicio: referido en Juniper como Clase de servicio, es una forma de manejar distintos perfiles de tráfico, priorizando unos sobre otros.
- CE: *Customer Edge,* equipo en una red WAN ubicado en el sitio final del suscriptor.
- Clase de servicio: referido por otros fabricantes como Calidad de servicio, es una forma de manejar distintos perfiles de tráfico, priorizando unos sobre otros.
- CLI: *Command Line Interface*, es la línea de commandos utilizada para operar y configurar dispositivos de red.
- Core: dentro de una red WAN, se refiere a dispositivos que cumplen con funciones de transmisión de servicios únicamente:
- CPE: *Customer Premises Equipment,* equipo en una red WAN ubicado en el sitio final del suscriptor.
- CSMA/CD: *Carrier Sense Multiple Access / Collision Detection*, es un algoritmo que determinar las acciones a tomar cuando dos dispositivos transmiten al mismo tiempo sobre un mismo medio.
- CSU/DSU: *Channel Service Unit/Data Service Unit*, dispositivo que transforma información de tecnología LAN a tecnología WAN y vice-versa.

D

- Datagramas: equivalente a paquetes, es la forma usual de referir datos en capa 3.
- Des-encapsulación: proceso de mover datos de una capa a una más alta.
- Distance-Vector: tipo de protocolo de ruteo dinámico que utiliza los saltos o distancias entre equipos como métrica para escoger la mejor ruta.
- Distancia administrativa: medida utilizada por equipos Cisco de capa 3 para discernir entre el uso de uno u otro protocolo.
- Dominio de Broadcast: conjunto de equipos de red dentro de una misma LAN o segmento de red.
- Dominio de Colisión: conjunto de equipos que comparten el mismo medio de transmisión.

E

- Eficacia: capacidad o factibilidad de cumplir con un objetivo.
- Eficiencia: cumplir a detalle un objetivo trazado.
- Encapsulación: mover información de una capa a otra más baja.

- Escalabilidad: capacidad de aplicar un objetivo a un nivel amplio o mayor, a gran escala.
- Estándar: modelo o referencia que sirve de guía al aplicarlo en un proceso.

F

- Fast Ethernet: serie de estándares de redes Ethernet de 100 Mbps.
- Full-Duplex: capacidad de transmitir y recibir información al mismo tiempo.

G

- Giga Ethernet: serie de estándares de redes Ethernet de 1000 Mbps.

H

- Half-Duplex: capacidad de transmitir o recibir información, pero nunca al mismo tiempo.
- ICMP: *Internet Control Message Protocol*, protocolo de capa 3 utilizado para el envío de mensajes producto del uso de otro protocolo de capa 3.

I

- Interfaz: en equipos de red, se refiere al puerto de salida para la comunicación con otros.

J

- Junos OS: sistema operativo en equipos Juniper.

L

- L2Circuit: protocolo en equipos Juniper que simula una conexión de capa 2 en un equipo de capa 3.
- Link-State: tipo de protocolo de ruteo dinámico que utiliza como métrica el estado de links.
- Loopback: interfaz virtual comúnmente utilizada para identificar un equipo y gestionarlo.
- Mensaje: denominación a datos de capa de aplicación.

M

- Máscara de subred: Valor de 32 bits, que señala qué bits de una dirección IPv4 corresponden a la identificación de la red, y qué porción al host.
- Métrica: método de protocolos de enrutamiento para asignar costos a los diferentes caminos a un punto remoto:
- Modelo OSI: modelo de referencia de 7 capas.
- Modelo TCP/IP: modelo aplicado de 4 capas que toma como referencia el Modelo OSI.

O

- Octeto: 8 bits.
- Open-source: software sin fines de lucro y al alcance público.

P

- Paquetes: denominación a información en capa 3.
- Parsing: término utilizado al analizar un conjunto de información, con el fin de obtener un elemento específico del mismo.
- Path-Vector: tipo de protocolo de ruteo dinámico que actualiza la información de la ruta hacia un destino.

- Política de ruteo: regla que especifica un conjunto de condiciones para importar o exportar rutas a través de un protocolo de ruteo dinámico en específico.
- POP: *Point of Presence,* es usualmente un equipo que representa una frontera en la red de un ISP.
- Preferencia de ruta: medida utilizada por equipos Juniper de capa 3 para discernir entre el uso de uno u otro protocolo.
- Prefijo: dirección IP que puede representar un conjunto de otras dependiendo de la máscara de subred aplicada.
- Protocolo: conjunto o sistema de reglas que permiten la comunicación entre dos o más entidades.
- Puerto: interfaz o salida de un equipo para la comunicación con otros.

R

- RFC: *Request for comments,* documento formal de IETF sobre protocolos o información de interés relacionada a las telecomunicaciones.
- Ruta activa: ruta o camino seleccionado de todos los disponibles para llegar a un destino.

S

- Script: programa o documento que permite la implementación automatizada de procesos o algoritmos.
- Sede: En VPLS, se refiere a un local final del suscriptor del servicio.
- Segmentos: denominación a datos de capa 4.
- Simplex: transmisión de información
- Síncrono: en telecomunicaciones, se refiere a la transmisión de información bajo la pauta de una señal de reloj externo.
- Sistema Lógico: en Juniper, se refiere a la partición de un equipo
- Sitio: en VPLS, término utilizado para referirse a un PE de la red MPLS.
- Spanning Tree Protocol: Método utilizado en switches para prevenir loops.
- SSH: Protocolo de gestión remota de equipos en forma segura (encriptada).

T

- Telecomunicaciones: transmisiones y recepciones de señales de cualquier naturaleza a través de las cuales se transmite cualquier tipo de información comunicada a distancia.
- Tramas: denominación de datos de capa 2.

V

- VLAN: red virtual en la que sus elementos comparten un mismo segmento de red sin estar físicamente cerca.

X

- XML: lenguaje de marcas utilizado para almacenar datos en forma legible.