Reconfiguración Automática de Parámetros de Calidad de Servicio Dispositivos por medio del Protocolo NETCONF

Ing. Javier A. Ouret
Facultad de Ingeniería
Carrrera de Ingeniería en Informática
Cátedra de Redes III
Universidad Católica Argentina
javierouret@uca.edu.ar

Ing. Ignacio Parravicini
Facultad de Ingeniería
Carrrera de Ingeniería en Informática
Cátedra de Redes III
Universidad Católica Argentina
ignacioparravicini@uca.edu.ar

Resumen. Estudio y desarrollo de un modelo de aprendizaje automático supervisado para los parámetros de la Calidad de Servicio (SLA) en enlaces de acceso en última milla. Los parámetros que alimentan el modelo son el resultado de las pruebas definidas por las metodologías estándar RFC2544 - UIT.Y1564. Se plantea el uso del protocolo NETCONF para configurar automáticamente los parámetros de tráfico necesarios para cumplir con los acuerdos de calidad de servicio. Al realizar pruebas disruptivas y no disruptivas para la integridad del servicio, un proveedor de servicios puede certificar que los parámetros de trabajo de un circuito Ethernet entregado cumplen con las expectativas del usuario final, evitando el rendimiento deficiente de las aplicaciones ejecutadas en ese circuito. El enfoque de aprendizaje automático supervisado se basa en el algoritmo K-NN de los vecinos más próximos (k-nearest neighbors). Encontramos que tanto la técnica supervisada como la no supervisada son excelentes herramientas para explorar y clasificar parámetros de servicio como retardo de trama, variación de retardo de trama, intervalos de pérdida de paquetes altos, disponibilidad y rendimiento. Se puede aplicar una correlación de parámetros con el tipo de servicio requerido para los flujos de red (IP en tiempo real para aplicaciones de datos, video y voz) y establecer automáticamente los perfiles de ancho de banda. Los perfiles de ancho de banda se pueden luego configurar por puerto, basados en VLAN y CoS, en uno o varios EVC (circuitos virtuales Ethernet) del dispositivo UNI. Para la implementación de la configuración, adoptamos el lenguaje de modelado de datos Yang y el protocolo de codificación de mensajes NETCONF de XML, seguidos de una activación orquestada en los dispositivos de red a través de múltiples transacciones NETCONF.

Palabras Clave: NETCONF, K-NN, expectation maximization, performance monitoring, RFC2544, Y.1564, machine learning, service level agreement.

1 Introducción

El crecimiento exponencial de Internet está haciendo ineficientes varios protocolos y métodos diseñados originalmente para monitorear y configurar los dispositivos de red (por ejemplo, demarcadores de servicio, NIDs, conmutadores y enrutadores). También se observa una evolución hacia la virtualización de estos dispositivos por medio del paradigma de Software Define Networks (SDN) y derivaciones como SD-WAN. Con este trabajo se propone un método para: a) evaluación de la performance del nivel de servicio en operación para enlaces de acceso de las redes de proveedores de servicios de Capa 2 (Carrier o Metro Ethernet), b) ejecutar el proceso correctivo de configuración sobre los dispositivos de diferentes proveedores. Se prueba la viabilidad de automatizar el proceso para obtener métricas utilizando técnicas de aprendizaje automático supervisado. En particular, para esta versión del trabajo, el enfoque se fija en 2 parámetros para propósitos de prueba: retardo y variación de retardo (delay y delay variation). Estos son parámetros críticos durante la configuración de los perfiles de ancho de banda para el tráfico en tiempo real. Los cambios en la latencia o los cambios en el ancho de banda afectan la cantidad de tiempo requerido para que tramas de datos viajen entre un equipo de acceso y un equipo de usuario (CPE - Customer Premise Equipment). Si bien la latencia no tiene un impacto directo en el ancho de banda muchos protocolos utilizan diferentes mecanismos de encolado de paquetes y de evaluación de ACKs para definir sus ventanas de transmisión, a mayor latencia mayor es el tiempo que tarda en recibirse un ACK, lo que influye en el tamaño de esa ventana. En [14] se ilustra que a través de modelos de colas reducir la latencia física es más eficiente que aumentar el ancho de banda para acortar el tiempo de respuesta (latencia basada en la congestión), especialmente cuando los dispositivos involucrados tienen una baja utilización.

Con respecto al método para recopilar información de las redes operativas utilizaremos el aprendizaje automático de los parámetros seleccionados. El aprendizaje automático se considera un campo de inteligencia artificial que utiliza técnicas estadísticas para que los sistemas informáticos puedan "aprender" (por ejemplo, mejorar progresivamente el rendimiento en una tarea específica) a partir de los datos [1]. Para que el proceso de aprendizaje sea autosostenido, necesitamos utilizar técnicas que puedan procesar la información de los enlaces de la red con una mínima intervención humana externa.

Para evaluar y obtener los datos necesarios para este modelo se adoptó el algoritmo K-NN de los vecinos más próximos (k-nearest neighbors). El objetivo es encontrar los grupos de datos disponibles de los enlaces operativos [2]. Los datos se presentan para su evaluación como una agrupación suave (los grupos pueden superponerse). Para el retardo d y el jitter j (variación de retardo), un modelo con distribuciones en dos o tres dimensiones es aceptable para esta primera etapa de prueba.

Uno de los beneficios tanto de K-NN es que producen parámetros válidos para la selección de grupos en cada iteración [3].

Este trabajo es una variante de la primera versión sobre el tema presentada en CACIDI 2018 [15].

2 NETCONF – Conceptos y Alcance

NETCONF es un protocolo de administración de red IETF que define un mecanismo simple a través del cual se puede administrar un dispositivo, recuperar información de datos de configuración así como cargar y manipular nuevos datos de configuración [4]. NETCONF es un protocolo orientado a la conexión, ya que establece una sesión autenticada y mantiene el orden de las transmisiones. NETCONF utiliza a YANG como lenguaje de modelado para definir la estructura, la sintaxis y la semántica de los datos que se pueden usar para las operaciones de NETCONF, incluida la configuración, los datos de estado, los RPC y las notificaciones. Esto permite que el dispositivo exponga una interfaz de programación de aplicaciones (API) completa y formal. Las aplicaciones pueden utilizar esta API para enviar y recibir conjuntos de datos de configuración completos y parciales. NETCONF utiliza una codificación de datos basada en lenguaje de marcado extensible (XML) para los datos de configuración, así como para los mensajes de protocolo. NETCONF permite al usuario realizar operaciones CRUD[1] en la configuración de dispositivos de red, utilizando codificación de datos XML como contenedor de transferencia de datos del protocolo (aunque tiene soporte para JSON[2]). Además de las operaciones CRUD, que se implementan en un paradigma de comunicaciones basado en RPC, NETCONF implementa notificaciones de eventos que se pueden configurar para recibir alertas del elemento de red, como las que generalmente se envían utilizando SNMP. A diferencia de SNMP, no hay bases de datos estáticas como las MIBs, en cambio, las capacidades de los dispositivos se intercambian cuando se establece la conexión [4]. NETCONF mejora la funcionalidad, agrega más funciones y aumenta la eficiencia del proceso de configuración de la red. Además, proporciona transacciones únicas para datos de configuración complejos, incorpora seguridad en el protocolo de transporte y es más fácil desarrollar nuevos servicios que con comandos CLI y SNMP. Una solicitud de cliente NETCONF se puede ejecutar usando XML API: Ver. Fig 1.

Con todas estas funciones, las operaciones CRUD (además de las notificaciones en tiempo real) hacen que los elementos de la red sean más dinámicos y flexibles, permitiendo que las configuraciones puedan ser modificadas por cualquier software de terceros, en función de cualquier criterio, que podría incluir cambios de configuración, basados en estudios de big data y técnicas de aprendizaje automático como las sugeridas aquí.

3 Métricas Normalizadas para QoS y SLA

Existen varias métricas para la evaluación de la calidad de servicio. Algunos de ellos siguen recomendaciones de organizaciones estándar o RFC. En este documento, nos centramos en las métricas del Y.1731, el RFC2544 y el UIT Y.1564, normalizados por el MEF (Metro Ethernet Forum) [5] [6] [7] [8], pero pueden utilizarse otros métodos para definirlas.

Las métricas de calidad de servicio presentadas en las Tablas 1 y 2 son indicadores que permiten la caracterización de enlaces a nivel de capa 2, a través de MAC MEPs de dispositivos UNI y NNI, y evalúan sus limitaciones para determinar el nivel de cumplimiento del operador con respecto a expectativas del usuario. Estas métricas serán los datos que alimenten el K-NN.

```
<?xml version="1.0"?>
<nc:rpc message-id="16" xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns="http://www.net2edge.com/firmwos:6.0.1:if_manager">
<nc:edit-config>
  <nc:target>
    <nc:running/>
  </nc:target>
  <nc:config>
    <configure>
      <__XML__MODE__exec_configure>
      <interface>
        <ethernet>
          <interface>1/1</interface>
            <__XML__MODE_if-ethernet>
              <__XML__MODE_if-eth-base>
                <description>
                  <desc_line>Netconf testing</desc_line>
                </description>
              </__XML__MODE_if-eth-base>
            </__XML__MODE_if-ethernet>
        </ethernet>
      </interface>
      </__XML__MODE__exec_configure>
                                                  </configure>
  </nc:config>
```

Fig. 1. Solicitud NETCONF

METRICA	**UNI_A-UNI_Z**	**UNI_A-UNI_Z**	Time Interval **(T)**	**Small Time Interval Δ**	**Percentile**
FD (ms)	≤ 120	≤ 160	24h	*2m*	93th
FDR (ms)	≤ 30	≤ 30	24h	*2m*	93th
FLR (% of frames)	≤ 0.12	≤ 0.12	24h	*2m*	NA

Tabla 1. Métricas normalizadas del MEF [9].

Calidad de Servicio	**CoS ID**	**Bandwidth Profile EVC & per CoS ID**	**Service Performance**
Real Time IP- IP Telephony-IP Video			
1	6,7	CIR > 0 EIR = 0	Latency < 5 ms Jitter < 1 ms Packet Loss. < 0.001%
Critical burst data applications críticos with low latency and low packet loss (like Cloud storage)			
2	4,5	CIR > 0 EIR <= UNI Speed	Latency < 5 ms Jitter = N/S Packet Loss. < 0.01%
Burst data applications that requires Bandwidth assurance			
3	3,4	CIR > 0 EIR <= UNI Speed	Latency < 15 ms Jitter = N/S Packet Loss. < 0.1%
Best effort services			
4	0,1,2	CIR = 0 EIR = UNI Speed	Latency < 30 ms Jitter = N/S Packet Loss. < 0.5%

Tabla2. Tipo de servicios sugeridos [10].

4 Metodo propuesto para evaluar retardo (delay) y variation de retardo (delay variation / jitter)

Las métricas de la capa 2 obtenidas mediante la implementación del estándar Y.1731/RFC 2544/Y.1564 superan la precisión de las mediciones de tráfico realizadas con los valores de ICMP, RTT, BB o BDP en la capa 3. Para obtener estos valores de métricas, se utiliza el OAM IEEE 802.3ah. El protocolo OAM se ejecuta en la subcapa MAC de Ethernet utilizando los MEP y la ID de MIP para identificar los puntos finales del enlace. El MEP representa el punto final de la entidad que se va a administrar (en general, un conmutador o puerto de enrutador) y el MIP es un punto intermedio. Para una explicación detallada ver [5]. Cada MEP se identifica mediante un valor numérico entero asociado con su dirección MAC y define las conexiones entre todos los MEP lineales y MIP (EPS) o los MEP en forma de anillo (ERPS). La configuración de los MEP debe realizarse manualmente en cada dispositivo de demarcación de servicio según la metodología sugerida por el MEF e implementada en el firmware de los dispositivos de demarcación. Para cada uno de los MEP / MIP, las tramas OAM 0x8902 circulan y es posible rastrear todas las conexiones (Link Trace) y generar patrones de señal de prueba de diferentes tamaños (bloques de 64 a 9600 bytes) con patrones de 0s, 1s o 10101010 , secuenciado o no. Estas señales de prueba se utilizan para obtener métricas de variación de retardo y retardo. Y.1731 y Y.1564 son metodologías de prueba de servicio no disruptivas y se pueden ejecutar con tráfico en servicio, mientras que RFC2544 es disruptivo.

5 Justificación para el uso de técnicas de aprendizaje automático.

Los proveedores y operadores de servicios de Internet pueden gestionar desde cientos hasta millones de enlaces de capa 2 con configuraciones punto a punto, punto a multipunto o multipunto a multipunto. Para que Metro-Ethernet sea más escalable, el proveedor de servicios puede usar esquemas de encapsulación MAC para el reenvío de tramas en el borde. La cantidad de direcciones MAC en tránsito requiere el uso de arquitecturas de pseudocable (MPLS-TP, MAC en MAC, etc.) como IEEE 802.1ah en los puentes de la red troncal del proveedor para admitir grandes cantidades de instancias de servicio en dispositivos de red [11]. Cada enlace puede tener varios circuitos virtuales de EVC (canales virtuales de Ethernet) con múltiples VLAN de servicio (S-VLAN) o VLAN de cliente (C-VLAN), cada uno con sus propios perfiles de ancho de banda para el tráfico sensible al retardo. Por lo tanto, es necesario evaluar métricas específicas para decidir si se necesitará una corrección de ancho de banda. La cantidad de datos disponibles para la evaluación justifica la implementación de la captura y evaluación de datos supervisados y no supervisados, detectando grupos de métricas aceptables y no aceptables. Los métodos de agrupación permiten aprender la mezcla de parámetros a partir de datos. K-NN se adopta para esta tarea y para obtener una clasificación verosímil en un problema donde la media y la varianza de los grupos de métricas están ocultas.

6 Análisis de datos utilizando K-NN.

Para realizar el análisis de datos utilizamos el algoritmo de k-NN (k-Nearest Neighbors) implementándolo para clasificar instancias similares no visibles de las métricas bajo análisis. Buscamos el mejor valor entre las K instancias más similares del parámetro a encontrar. Para definir la similaridad utilizamos una métrica que mida la distancia entre dos puntos. La métrica de similaridad puede ser la distancia Euclideana u otras opciones como Manhattan, Chebyshev o Hamming. [18]

$$d(x, x') = \sqrt{(x_1 - x_1')^2 + (x_2 - x_2')^2 \ldots + (x_n - x_n')^2}$$

$$K = entero\ positivo; x = parámetro\ a\ encontrar; d = métrica\ de\ similaridad$$

K-NN se ejecuta sobre la información de tráfico obtenida computando d entre x y cada dato observado. Definimos al conjunto A como a los K puntos de los datos observados cercanos a x. K sew recomienda que sea impar para impedir una situación en la que dos o más participantes en un competencia se asignen por igual.

A continuación estimamos la probabilidad condicional para cada clase o fracción de puntos en A con una misma clase indicada.

$$I(x) = función\ clasificadora$$
$$1\ si\ x\ es\ vedadero - 0\ si\ x\ es\ falso$$
$$P(y = j|X = x) = \frac{1}{K}\sum_{i \epsilon A} I(y^{(i)} = j)$$

Finalmente x (como dato) es asignado a la clase que tenga la probabilidad más alta. K-NN realiza la búsqueda dentro del grupo de datos observados para las K instancias que más se parezcan a cada nueva instancia, asignando a ella la clase más común del grupo. Esto lleva a la definición o cálculo de una frontera o borde de decisión.

Uno de los problemas de este método radica en cómo seleccionar K, por lo que debemos evaluar los efectos de K dentro del clasificador K-NN. Se dice que K es un "hiperparámetro" que controla la forma de la frontera o borde de decisión. K en K-NN representa el número de datos vecinos que votan para determinar la posición de un nuevo caso observado. Cuanto más pequeño sea K más limitada será la distribución de los grupos de datos, haciendo que el resultado sea muy sesgado. Conviene seleccionar K impares para el caso de problemas con 2 clases. El principal problema de K-NN radica en la complexidad de su implementación ya que para que funcione necesitamos muestras de referencia dentro de los grupos correctos, para poder predecir otros grupos en el futuro. K-NN es supervisado pues tratamos de clasificar un parámetro observado en base a clasificaciones conocidas de otros puntos. Si aplicásemos una técnica del tipo K-Mean (EM) tomaría-mos un conjunto de datos no clasificados para tratar en agruparlos en manojos y sería no supervisada pues esos datos no devienen de clasificaciones externa previas. K en K-Mean determina el número de agrupaciones con la cuáles queremos terminar.

En el caso que tengamos datos discretos debemos convertirlos a numéricos. Para nuestro caso en particular tenemos variables independientes en los datos de entrenamiento que se miden en diferentes unidades, por lo que es importante estandarizar las variables antes de calcular las distancias [16]. Utilizamos el siguiente método de estandarización:

$$x_s = \frac{x - promedio}{desviación\ estándard}$$

Luego proponemos a modo de ejemplo una observación no clasificada (*NOTA: pueden existir miles de ellas, ésta es sólo para ejemplificar un caso*):

	Calidad	Retardo	Variación de retardo
Sin estandarizar	1.00	4.00	3.00
Estandarizado	-1.24	-0.90	0.59

Tabla 3. Parámetro observado a clasificar

Las características (features) retardo y variación de retardo, nos definen la calidad de servicio. NETCONF se utiliza en la ejecución de comandos para la recopilación de datos que alimentan a K-NN y, posteriormente, para la reconfiguración del perfil de ancho de banda del dispositivo que lo haga cumplir con el SLA, si es posible para la infraestructura definida en la red. A continuación en la Tabla 4. mostramos una porción de muestras obtenidas a modo de ejemplo con la clasificación de tipo de servicio 1 a 4 indicada en la tabla 2, los valores de retardo, retardo esperado, variación de retardo y el esperado, pérdida de paquetes, si cunmple el SLA, las distancias calculadas, K y orden. Se toma K=3 para el parámetro observado a clasificar (Tabla 3.). Al final de la tabla pueden observarse los promedios y DS necesarios para la estandarización aplicada en la Tabla. 5.

Se indican algunos valores de la muestra (27) a modo de ejemplo. El número total de muestras del conjunto de datos es de 987. Para el entrenamiento dividimos los datos entre un conjunto sobre el cual kNN hace predicciones y otro conjunto de prueba o testeo que usamos para evaluar la precisión del modelo. La división es aleatoria y usamos un valor para establecer la proporción o relación de esa división (variable “split”). Ver en [21]. En el ejemplo usamos un valor de “split” de 0.60. Se realiza el testeo y entrenamiento sobre el mismo conjunto de datos (dataset). El valor de K empleado es 3.

Parámetros													
Muestra	Ancho de Banda B (Mbps)	Tipo de Servicio	Calidad	Retardo (ms)	Valor Esperado (ms)	Variación de retardo (ms)	Valor esperado (ms)	Pérdida Paquetes < a	Valor esperado < a	Cumple	Distancia	K	Orden
1	768.87	CIR > 0 EIR = 0	1	5	5	2.58	1	0.001%	0.001%	NO	1.08	1	0.42
2	111.45	CIR > 0 EIR <= UNI Speed	2	4	5	2.00	5	0.01%	0.01%	SI	1.41	2	1.41
3	153.40	CIR > 0 EIR <= UNI Speed	3	12	15	4.31	4	0.1%	0.1%	SI	8.35	3	2.00
4	775.33	CIR = 0 EIR = UNI Speed	4	26	30	2.10	2	0.5%	0.5%	SI	22.22	4	2.24
5	861.76	CIR > 0 EIR <= UNI Speed	2	6	5	2.00	5	0.01%	0.01%	NO	2.45		2.24
6	423.88	CIR > 0 EIR = 0	1	9	5	2.58	1	0.001%	0.001%	NO	5.02		2.45
7	554.66	CIR > 0 EIR <= UNI Speed	3	17	15	4.31	4	0.1%	0.1%	NO	13.22		2.45
8	975.15	CIR > 0 EIR <= UNI Speed	2	6	5	2.00	5	0.01%	0.01%	NO	2.45		2.45
9	814.01	CIR > 0 EIR = 0	1	5	5	1.00	1	0.001%	0.001%	SI	2.24		2.45
10	50.36	CIR > 0 EIR <= UNI Speed	3	12	15	4.31	4	0.1%	0.1%	SI	8.35		2.45
11	429.50	CIR > 0 EIR <= UNI Speed	3	14	15	4.31	4	0.1%	0.1%	SI	10.28		2.45
12	554.71	CIR > 0 EIR <= UNI Speed	2	6	5	2.00	5	0.01%	0.01%	NO	2.45		2.45
13	766.09	CIR > 0 EIR = 0	1	9	5	2.58	1	0.001%	0.001%	NO	5.02		2.45
14	991.62	CIR = 0 EIR = UNI Speed	4	23	30	2.10	2	0.5%	0.5%	SI	19.26		2.45
15	503.81	CIR > 0 EIR = 0	1	4	5	1.00	1	0.001%	0.001%	SI	2.00		4.02
16	695.02	CIR > 0 EIR <= UNI Speed	2	6	5	2.00	5	0.01%	0.01%	NO	2.45		5.02
17	944.26	CIR > 0 EIR <= UNI Speed	2	6	5	2.00	5	0.01%	0.01%	NO	2.45		5.02
18	531.81	CIR > 0 EIR <= UNI Speed	2	6	5	2.00	5	0.01%	0.01%	NO	2.45		8.35
19	237.74	CIR > 0 EIR <= UNI Speed	3	14	15	4.31	4	0.1%	0.1%	SI	10.28		8.35
20	387.91	CIR > 0 EIR = 0	1	4	5	2.58	1	0.001%	0.001%	NO	0.42		10.28
21	301.40	CIR > 0 EIR <= UNI Speed	2	6	5	2.00	5	0.01%	0.01%	NO	2.45		10.28
22	559.43	CIR = 0 EIR = UNI Speed	4	29	30	2.10	2	0.5%	0.5%	SI	25.20		13.22
23	948.50	CIR > 0 EIR <= UNI Speed	2	6	5	2.00	5	0.01%	0.01%	NO	2.45		18.27
24	485.66	CIR = 0 EIR = UNI Speed	4	27	30	2.10	2	0.5%	0.5%	SI	23.21		19.26
25	265.84	CIR > 0 EIR = 0	1	5	5	1.00	1	0.001%	0.001%	SI	2.24		22.22
26	216.72	CIR > 0 EIR <= UNI Speed	2	6	5	2.00	5	0.01%	0.01%	NO	2.45		23.21
27	749.62	CIR = 0 EIR = UNI Speed	4	22	30	2.10	2	0.5%	0.5%	SI	18.27		25.20
		Promedio	2.30	10.93	11.48	2.42	3.30						
		DS	1.05	7.73	9.60	0.99	1.70						

Tabla 4. K-NN aplicado sobre algunos datos de muestra antes de la estandarización.

En la Tabla 5. mostramos lo mismo que en la tabla 4 pero con valores estandarizados, que son los utilizados efectivamente en el modelo.

Parámetros Estandarizados													
Muestra	Ancho de Banda B (Mbps)	Tipo de Servicio	Calidad	Retardo (ms)	Valor Esperado (ms)	Variación de retardo (ms)	Valor esperado (ms)	Pérdida Paquetes < a	Valor esperado < a	Cumple	Distancia	K	
1	768.87	CIR > 0 EIR = 0	-1.24	-0.77	-0.67	0.16	-1.35	0.001%	0.001%	NO	0.44	1	0.43
2	111.45	CIR > 0 EIR <= UNI Speed	-0.28	-0.90	-0.67	-0.43	1.00	0.01%	0.01%	SI	1.39	2	0.44
3	153.40	CIR > 0 EIR <= UNI Speed	0.67	0.14	0.37	1.91	1.91	0.1%	0.1%	SI	2.55	3	0.77
4	775.33	CIR = 0 EIR = UNI Speed	1.63	1.95	1.93	-0.33	-0.33	0.5%	0.5%	NO	4.14	4	0.77
5	861.76	CIR > 0 EIR <= UNI Speed	-0.28	-0.64	-0.67	-0.43	1.00	0.01%	0.01%	NO	1.42		1.39
6	423.88	CIR > 0 EIR = 0	-1.24	-0.25	-0.67	0.16	-1.35	0.001%	0.001%	NO	0.77		1.42
7	554.66	CIR > 0 EIR <= UNI Speed	0.67	0.79	0.37	1.91	1.91	0.1%	0.1%	NO	2.87		1.42
8	975.15	CIR > 0 EIR <= UNI Speed	-0.28	-0.64	-0.67	-0.43	1.00	0.01%	0.01%	NO	1.42		1.42
9	814.01	CIR > 0 EIR = 0	-1.24	-0.77	-0.67	-1.44	-1.35	0.001%	0.001%	SI	2.03		1.42
10	50.36	CIR > 0 EIR <= UNI Speed	0.67	0.14	0.37	1.91	1.91	0.1%	0.1%	SI	2.55		1.42
11	429.50	CIR > 0 EIR <= UNI Speed	0.67	0.40	0.37	1.91	1.91	0.1%	0.1%	NO	2.66		1.42
12	554.71	CIR > 0 EIR <= UNI Speed	-0.28	-0.64	-0.67	-0.43	1.00	0.01%	0.01%	NO	1.42		1.42
13	766.09	CIR > 0 EIR = 0	-1.24	-0.25	-0.67	0.16	-1.35	0.001%	0.001%	NO	0.77		1.42
14	991.62	CIR = 0 EIR = UNI Speed	1.63	1.56	1.93	-0.33	-0.33	0.5%	0.5%	SI	3.88		1.42
15	503.81	CIR > 0 EIR = 0	-1.24	-0.90	-0.67	-1.44	-1.35	0.001%	0.001%	SI	2.03		2.03
16	695.02	CIR > 0 EIR <= UNI Speed	-0.28	-0.64	-0.67	-0.43	1.00	0.01%	0.01%	NO	1.42		2.03
17	944.26	CIR > 0 EIR <= UNI Speed	-0.28	-0.64	-0.67	-0.43	1.00	0.01%	0.01%	NO	1.42		2.03
18	531.81	CIR > 0 EIR <= UNI Speed	-0.28	-0.64	-0.67	-0.43	1.00	0.01%	0.01%	NO	1.42		2.55
19	237.74	CIR > 0 EIR <= UNI Speed	0.67	0.40	0.37	1.91	1.91	0.1%	0.1%	NO	2.66		2.55
20	387.91	CIR > 0 EIR = 0	-1.24	-0.90	-0.67	0.16	-1.35	0.001%	0.001%	NO	0.43		2.66
21	301.40	CIR > 0 EIR <= UNI Speed	-0.28	-0.64	-0.67	-0.43	1.00	0.01%	0.01%	NO	1.42		2.66
22	559.43	CIR = 0 EIR = UNI Speed	1.63	2.34	1.93	-0.33	-0.33	0.5%	0.5%	NO	4.42		2.87
23	948.50	CIR > 0 EIR <= UNI Speed	-0.28	-0.64	-0.67	-0.43	1.00	0.01%	0.01%	NO	1.42		3.80
24	485.66	CIR = 0 EIR = UNI Speed	1.63	2.08	1.93	-0.33	-0.33	0.5%	0.5%	NO	4.23		3.88
25	265.84	CIR > 0 EIR = 0	-1.24	-0.77	-0.67	-1.44	-1.35	0.001%	0.001%	SI	2.03		4.14
26	216.72	CIR > 0 EIR <= UNI Speed	-0.28	-0.64	-0.67	-0.43	1.00	0.01%	0.01%	NO	1.42		4.23
27	749.62	CIR = 0 EIR = UNI Speed	1.63	1.43	1.93	-0.33	-0.33	0.5%	0.5%	SI	3.80		4.42

Tabla 5. K-NN aplicado sobre algunos datos de muestra luego de la estandarización.

En la Fig. 2 pueden observarse gráficamente los grupos aprendidos. El eje horizontal (x) indica los valores normalizados o estandarizados Retardo y Variación de Retardo, el eje y la Calidad del servicio.

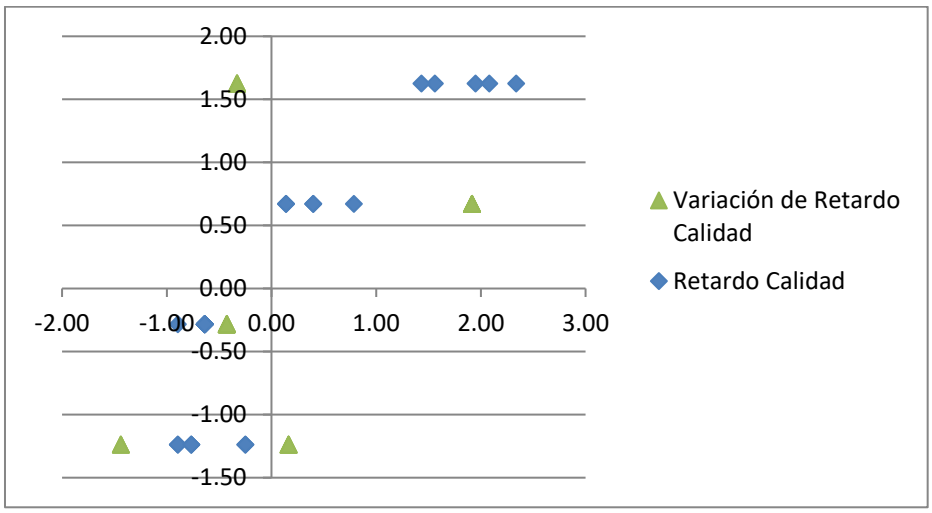

Fig.2 . K-NN Grupo de datos sobre muestra luego de la estandarización.

7 Configuración para obtener Retardos y Variación de retardos MEP DM (Delay Measurement)

La medición del retraso se realiza entre los MEPs. Pueden existir muchos MEP en un grupo, pero el DM (Medición de Retardo) se realiza entre dos MEP sólo porque es un punto de flujo hacia una función multipunto. Tanto el retraso unidireccional como el bidireccional + la variación del retardo pueden calcularse en función de la información intercambiada entre los MEPs.

```
(config)# mep 1 dm 1 dual flow multi interval 10
last-n
mep = mep command.
1 = mep ID.
dm = delay measurement.
1 = Priority in case of tagged OAM. In the MPLS
and EVC domain this is the COS-ID.
dual = Delay Measurement based on 1DM PDU
transmission.
flow =  The two way delay is calculated as round
trip symmetrical flow delay. The far end residence
time is subtracted.
multi = OAM PDU is transmitted with multicast MAC.
interval 10 = between PDU transmission in ms.
last-n = delay for average last N calculation min
value is 10.
```

```
(config)# perf-mon interval dm 15
(config)# do show mep dm

MEP DM state is:
RxTime : Rx Timeout
RxErr : Rx Error
AvTot : Average delay Total
AvN : Average delay last N
Min : Min Delay value
Max : Max Delay value
AvVarT : Average delay Variation Total
AvVarN : Average delay Variation last N
MinVar : Min Delay Variation value
MaxVar : Max Delay Variation value
```

Fig. 3. Comandos CLI para la medición de DM.

La Fig. 3 muestra la versión del comando CLI para la medición de retardo que se convertirá en una sesión de configuración de NETCONF XML y el comando para ejecutar la prueba y el comando para recordar los resultados. Toda la información de retraso y variación de retraso se envía a un almacenamiento de datos para un análisis posterior con el algoritmo K-NN, junto con la identificación de cada dispositivo (IP), puerto, VLAN, EVC y perfil de ancho de banda. La Fig. 4 presenta la versión CLI de la Fig. 3 convertida a entrada XML para NETCONF.

Configuration Request

```
<?xml version="1.0"?>
<nc:rpc message-id="16" xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns="http://www.net2edge.com/firmwos:6.0.1:if_manager">
  <nc:edit-config>
    <nc:target>
      <nc:running/>
    </nc:target>
    <nc:config>
      <configure>
        <__XML__MODE__exec_configure>
        <mep>
          <mep-id> 1 </mep-id>
          <delay measurement> dm </delay measurement>
          <priority> 1 </priority>
          <flow> dual flow </flow>
          <OAM PDU> multi </OAM PDU>
          <interval> interval 10 </interval>
          <last n delay > 10  </last n delay>
```

Fig. 4. Comandos CLI en versión NETCONF.

8 Modelo de Implementación

Para propósitos de prueba se evaluó inicialmente la plataforma Testtool. La herramienta permite simular hasta 1000 dispositivos NETCONF y hacer pruebas de escala. Se basa en una implementación central del servidor Netconf desde el controlador OpenDayLight ODL [13]. El algoritmo K-NN se programó utilizando Python y se probó con los datos obtenidos utilizando la metodología explicada. El programa en Python utilizado fue codificado por los autores para este problema en particular tomado como código base el descripto en [20]. Todavía no ha sido optimizado a nivel de programación pero funciona correctamente y puede ser descargado desde [21]. Los archivos de prueba pueden ser descargados de [22]. Las fuentes de datos fueron enlaces de la capa 2 de Lab con el tráfico generado por las herramientas RFC2544 y Y.1564. Para validar los parámetros en un entorno real, utilizamos los NIDs Liberator 304, 306 y 4424 Carrier Ethernet de Net2Edge (UK). Posteriormente utilizó NETCONF [4] y OpenWRT [19]. Se configuró una máquina virtual con Freenetconfd, y se realizaron pruebas de visualización de parámetros. El mismo requiere una recompilación de OpenWRT x86/64 para agregar la funcionalidad. Una alternativa en evaluación es utilizar módulos Docker. OpenWRT resulta una plataforma admisible como cliente NETCONF para impactar las nuevas configuraciones devenidas del proceso de toma de decisiones del algoritmo EM. En el proceso de investigación, encontramos un protocolo que afirma ser la evolución de NETCONF, definido en la RFC8040. Se requiere un estudio adicional del protocolo para terminar de comprender las ventajas respecto a NETCONF, y su factibilidad de implementación en las siguientes etapas del proyecto, a los efectos de compararlos. En la Fig.5 puede observarse el diagrama de bloques del modelo de implementación.

El modelo de implementación cuenta con un enrutador virtual corriendo OpenWRT x86, el cuál puede ser configurado mediante CLI o web a través de UCI (Unified Configuration Interface) o LUCI (LUA Unified Configuration Interface)r espectivamente. Adicionalmente es posible entrar al Shell del sistema operativo Linux, a través de ASH (BASH de ChromiumOS). Estas interfaces son necesarias sólo para la configuración inicial, ya que luego las actualizaciones pueden realizarse a través de Netconf, como se explica a continuación. Freenetconfd es el Daemon encargado de la interacción Netconf con dispositivos externos. Es el proceso que recibe y envía los archivos XML tanto con la configuración a impactar, como con la información reportada por el dispositivo.

El módulo MAND es el encargado de administrar los datos tanto de configuración del dispositivo, como de desempeño, pudiendo persistir dicha configuración utilizando el Agente de Configuración, conectado mediante un IPC. A su vez, es quien contiene el modelo YANG, su API, y un módulo para serializar y de-serializar las comunicaciones XML. Se comunica con el servidor Netconf mediante otro IPC que lo conecta con el Daemon freenetconfd.

El sistema de autoconfiguración, que actúa como servidor Netconf, es el encargado de realizar las consultas utilizando la API y el modelo YANG del dispositivo virtual, alimentar el algoritmo K-NN, y luego del proceso de clasificación de estado de puertos, decidir cuáles son las modificaciones a la configuración necesarias, e impactarlas en el dispositivo utilizando el mismo camino a la inversa, es decir, serializando un XML con la nueva configuración y enviándolo al daemon freenetconfd del enrutador virtual.

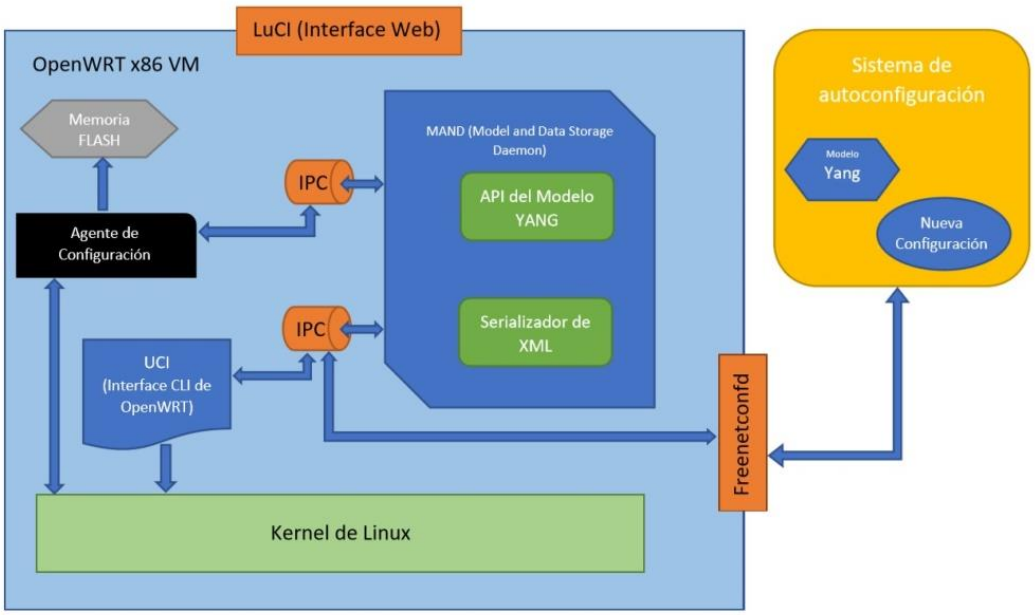

Fig. 5. Modelo de Implementación

9 Conclusiones

La metodología propuesta en este documento se utiliza para aprender y clasificar grupos de valores de métricas durante la operación de los enlaces de la capa 2 de Carrier Ethernet. Los valores aprendidos se utilizan para evaluar los distintos enlaces y decidir si se necesita realizar un ajuste de los perfiles de ancho de banda en los dispositivos. Se utiliza K-NN para clasificar los datos y para saber si un dato observado requiere que se aplique el protocolo NETCONF para corregir la configuración. Como resultado, se demuestra que se reducen la intervención humana necesaria para recopilar, analizar y seleccionar datos métricos. Como resultado colateral es de esperarse que la cantidad de errores del proceso también disminuya. El proceso de reconfiguración de los perfiles de ancho de banda recibe el resultado de las clasificaciones del modelo K-NN y se genera una base de datos histórica para el estudio del rendimiento del tráfico de la capa de enlace..

10 Trabajos Futuros

En la siguiente evolución de este trabajo haremos la expansión hacia una técnica combinada entre K-NN y la Maximización de Expectativas (EM) usando Modelos Mezclas Gaussianas (Gaussian Mixtures Model). K-NN será utilizado como un preprocesador de EM GMM para reducir la cantidad de datos de entrenamiento. K-NN es una técnica de clasificación mientras que GMM es una técnica de agrupamiento (clustering). El modelo futuro sería de tipo mixto, por cada grupo decimos que tiene una distribución probabilística y los parámetros (por aprender o descubrir) son obviamente desconocidos (media μ y varianza σ). EM se puede usar para generar la mejor hipótesis para los parámetros de distribución de las métricas de MEF seleccionadas, de modo de maximizar la probabilidad que los datos que buscamos provengan de distribuciones K, cada una con una media μ y una varianza σ^2. El objetivo es ingresar las métricas recopiladas en EM para realizar también un agrupamiento suave como con K-NN. Cada grupo de métricas corresponde a una distribución de probabilidad y desea encontrar sus parámetros (media y varianza). El algoritmo EM permite derivar estos parámetros. Algunos valores de retardo (y de variación de retardo) pueden coincidir con el SLA requerido (por ejemplo: pertenecen al grupo *r1*) y otros no (por ejemplo: si pertenecen al grupo *r2*) pero la membresía no está definida en el conjunto de datos disponibles. Como (μ_{r1},σ_{r1}^2), (μ_{r2},σ_{r2}^2) son necesarios para adivinar la fuente de datos y la fuente es necesaria para (μ_{r1},σ_{r1}^2), (μ_{r2},σ_{r2}^2) el algoritmo de EM ayuda a resolver este problema. Esto se conoce como una estimación de probabilidad máxima. Dados algunos datos, calculamos el valor de los parámetros que mejor explican esos datos. En el futuro trabajo los datos de prueba serán obtenidos a partir de grupos de valores de retardo (***r1***) y de valores de variación de retardos (***vr1***). A continuación se muestran las ecuaciones para ***vr1*** pero pueden ser fácilmente extendidas para ***vr1***. Se inicia el EM con distribuciones repartidas en forma aleatoria. (μ_{r1},σ_{r1}^2).

Para cada $P(r1|x_i)$ evaluamos si viene del grupo r1 o no y ajustamos (μ_{r1},σ_{r1}^2), para que se vinculen con los puntos asociados al grupo.

$$Estimación\ \mu_{r1} = \ \mu_{r1} \cong \frac{\sum(x_i)}{n_{r1}}$$

$$Estimated\ \sigma_{r1} = \ \sigma_{r1} \cong \frac{\sum(x_i - \mu_{r1}\sim)^2}{n_{r1}}$$

$$f(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \ .e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Probabilidad que x_i pertenezca al grupo de retardos r1 (cumpliendo con los expectativas de las métricas de SLA para retardos).

$$P(x_i|r1) = \frac{1}{\sqrt{2\pi\sigma_{r1}^2}} e^{-\frac{(x_i-\mu_{r1})^2}{2\sigma_{r1}^2}}$$

Para cada x_i tratamos de encontrar de cuál conjunto de parámetros (μ,σ^2) tiene su origen: $((\mu_{ri},\sigma_{ri}^2)$. Se realiza una asignación liviana basada en la probabilidad y no en el valor en si mismo:

$$r1_i = P(r1|x_i) = \frac{P(x_i|r1)P(r1)}{P(x_i|r1)P(r1) + P(x_i|r2)P(r2)}$$

$$r2_i = P(r2|x_i) = 1 - r1_i$$

Luego los (μ,σ^2) son recalculados para mejorar la asignación de valores:

$$\mu_{r1} = \frac{r1_1x_1 + r1_2x_2 + \cdots + r1_nx_n}{r1_1 + r1_2 + \cdots + r1_n}$$

$$\sigma_{r1}^2 = \frac{r1(x_1 - \mu_{r1})^2 + 1(x_2 - \mu_{r1})^2 + \ldots + r1(x_n - \mu_{r1})^2}{r1_1 + r1_2 + \cdots + r1_n}$$

Si tenemos otro grupo de retardos r2:

$$\mu_{rd2} = \frac{r2_1x_1 + r2_2x_2 + \cdots + r2_nx_n}{r2_1 + r2 + \cdots + r2_n}$$

$$\sigma_{r2}^2 = \frac{r2(x_1 - \mu_{r2})^2 + r2(x_2 - \mu_{r2})^2 + \ldots + r2(x_n - \mu_{r2})^2}{r2_1 + r2_2 + \cdots + r2_n}$$

$$P(r1) = \frac{r1_1 + r1_2 + \cdots + r1_n}{n}$$

$$P(r2) = 1 - P(r1)$$

La iteración continúa hasta la convergencia y extrae la media y la varianza para cada grupo de métricas. A partir de aquí, decida a qué grupo debe reconfigurarse para adaptarse mejor al SLA requerido. Ejemplo:

```
r1/r2 set 1: [μ=[ 2.036 54.479], σ=[ 0.069  0.435]
r1/r2 set 2: [μ=[ 4.29  79.968], σ=[[ 0.17   0.941]
```

NETCONF se utilizará para la recopilación de datos que alimentan a K-NN y EM-GMM y, posteriormente, para la reconfiguración del perfil de ancho de banda del dispositivo que lo haga cumplir con el SLA, si es posible para la infraestructura definida en la red.

11 Referencias

[1] Samuel, Arthur. "Some Studies in Machine Learning UsiDisponibleng the Game of Checkers", IBM Journal of Research and Development, 1959.

[2] Dempster, A.P.; Laird, N.M.; Rubin, D.B. "Maximum Likelihood from Incomplete Data via the EM Algorithm". Journal of the Royal Statistical Society, Series B. 39, 1977, pp 1–38.

[3] Padhraic Smyth, “Mixture Models and the EM Algorithm”, Department of Computer Science, University of California, Irvine, 2017

[4] R. Enns, M. Bjorklund, J. Schoenwaelder, A. Bierman, RFC 6241 “Network Configuration Protocol (NETCONF)” , IETF, 2011.

[5] Y.1731. “Operation, Administration And Maintenance OAM”, Functions and mechanisms for Ethernet based networks, ITU, 2006.

[6] S. Bradner,J. McQuaid, RFC 2544 “Benchmarking Methodology for Network Interconnect Devices”, IETF, 1999. Disponible: https://tools.ietf.org/html/rfc2544.

[7] ITU Y.1564. Ethernet service activation test methodology. ITU 2016.
[8] Christopher Cullan, Understanding Carrier Ethernet Service Assurance. Part I & II. MEF. 2016.
[9] MEF Performance Monitoring Metrics.
Available: https://wiki.mef.net/pages/viewpage.action?pageId=24710765
[10] MEF Technical Specification 6.2 “EVC Ethernet Services Definitions”, MEF, 2011.
[11] IEEE 802.1ah - Provider Backbone Bridges.
Disponible: http://www.ieee802.org/1/pages/802.1ah.html
[12] Liberator 4424 CLI Configuration Guide, Net2Edge Ltd, Rev.D, 2017.
[13] OpenDayLight
Disponible: https://www.opendaylight.org/
[14] Bandwidth and Latency: Their Changing Impact on Performance. Yiping Ding. BMC Software. 2005.
[15] Quality of Service Assesment Using Machine Learning Techniques for the NETCONF Protocol”. Javier Ouret, Ignacio Parravicini. CACIDI 2018.
[16] Assumptions of KNN- Standardization
Disponible:https://www.r-bloggers.com/k-nearest-neighbor-step-by-step-tutorial/
[17] Application of the Weighted K-Nearest Neighbor Algorithm for Short-Term Load Forecasting. Guo-Feng Fan 1, Yan-Hui Guo , Jia-Mei Zheng and Wei-Chiang Hong. Energies 2019, 12, 916; doi:10.3390/en12050916.
[18] A Complete Guide to K-Nearest-Neighbors with Applications in Python and R
Disponible: https://kevinzakka.github.io/2016/07/13/k-nearest-neighbor/
[19] OpenWRT
Disponible: https://openwrt.org/
[20] Machine Learning Algorithms - Jason Brownlee A Complete Guide to K-Nearest-Neighbors with Applications in Python and R.
Disponible:https://machinelearningmastery.com/tutorial-to-implement-k-nearest-neighbors-in-python-from-scratch/
[21] Código Python para k-NN
Disponible: https://drive.google.com/file/d/1xIXeSGIcUxATErtmxPvyWjwJyRgDe4MF/view?usp=sharing
[22] Archivo de prueba utlizado para el algoritmo k-NN
Disponible: https://drive.google.com/file/d/1Ewl8JqFEl6yzgE4DC0XLc4Xt7ubzgwMa/view?usp=sharing