

Monitorización de tráfico en redes WAN definidas por software mediante controlador RyU

Luis Leonardo Bascopé Aparicio

Andrés Jorge Muracciole Vázquez

Miniproyectos – 2019/2020



POLITÉCNICA

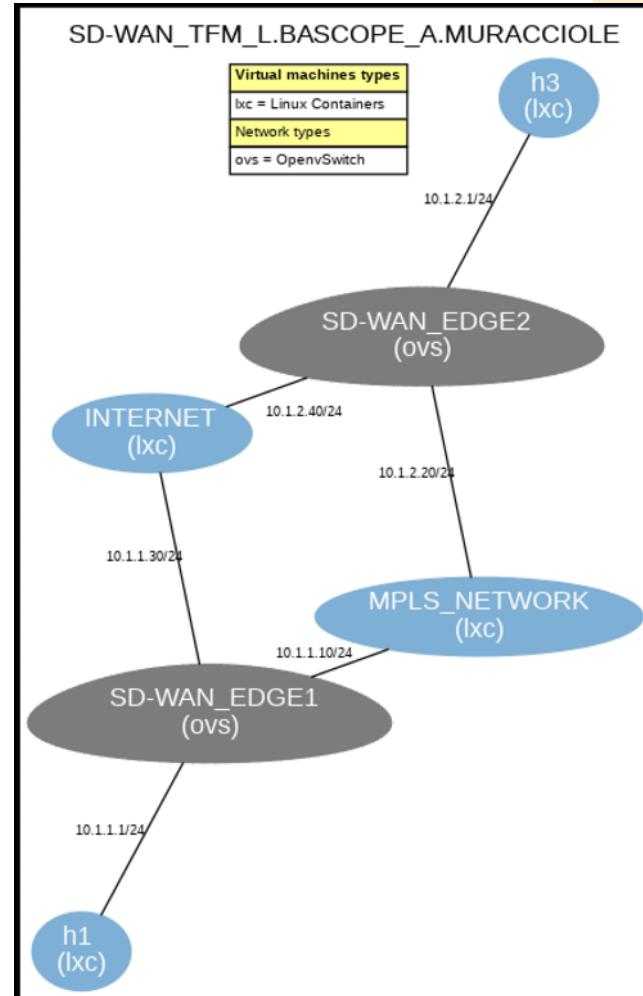
UNIVERSIDAD
POLÍTÉCNICA
DE MADRID

Objetivos Principales

- Desarrollar un escenario de red con la herramienta de virtualización de redes Virtual Network over LinuX (VNX) que emule una red WAN definida por software (SD-WAN) gestionada por el controlador RYU.
- Monitorizar el tráfico de la red SD WAN simulada en VNX, por medio del controlador RYU, utilizando “Traffic Monitor” y a través de este conseguir estadísticas que describan el comportamiento de la red.
- Analizar las aplicaciones de RYU existentes que permiten extraer de la red SDN métricas de calidad de servicio.
- Analizar los parámetros de calidad de servicio de Latencia, Ancho de Banda y perdida de paquetes, en el tráfico del escenario

Escenario

- Hosts
- Conmutadores (SD-WAN_EDGE1 y SD-WAN_EDGE2)
- Redes “INTERNET” y “MPLS”
- Controlador RyU



BANDWIDTH

- Cálculo de ancho de banda libre en cada puerto de los commutadores

Bandwidth

- Se adaptó el código original de Traffic Monitor para poder calcular el ancho de banda de los canales.
- Se calcula el uso instantáneo del canal y se le resta al ancho de banda total, obteniendo así el ancho de banda libre.
- Se adaptó el canal a 100 Mbps mediante “rest_qos.py”.

Modelado del BW con “rest_qos.py”

Para ejecutar exitosamente esta aplicación, se necesitan dos datos principales, los cuales son:

- Datapath ID (dpid) de cada OVS y
- Nombre de puerto (port_name) de cada interfaz.

Datapath ID

```
[QoS][INFO] dpid=000000000000000700: Join qos switch.  
[QoS][INFO] dpid=000000000000000800: Join qos switch.
```

Port Name:

```
$ sudo ovs-vsctl show
```

```
Bridge "SD-WAN_EDGE2"  
    Controller "tcp:127.0.0.1:6633"  
        is_connected: true  
    fail_mode: secure  
    Port "h2-e1"  
        Interface "h2-e1"  
    Port "SD-WAN_EDGE2"  
        Interface "SD-WAN_EDGE2"  
            type: internal  
    Port "MPLS_NETWORK-e2"  
        Interface "MPLS_NETWORK-e2"  
    Port "INTERNET-e2"  
        Interface "INTERNET-e2"  
Bridge "SD-WAN_EDGE1"  
    Controller "tcp:127.0.0.1:6633"  
        is_connected: true  
    fail_mode: secure
```

```
fail_mode: secure  
Port "SD-WAN_EDGE1"  
    Interface "SD-WAN_EDGE1"  
        type: internal  
Port "MPLS_NETWORK-e1"  
    Interface "MPLS_NETWORK-e1"  
Port "INTERNET-e1"  
    Interface "INTERNET-e1"  
    Port "h1-e1"  
        Interface "h1-e1"  
ovs version: "2.9.5"
```

Configurar la dirección de la base datos de cada OVS o “ovsdb_addr:

```
curl -X PUT -d '"tcp:127.0.0.1:6632"'  
http://localhost:8080/v1.0/conf/switches/00000000000000700/ovsdb_addr
```

```
curl -X PUT -d '"tcp:127.0.0.1:6632"'  
http://localhost:8080/v1.0/conf/switches/00000000000000800/ovsdb_addr
```

Configurar las colas para cada uno de los Switches

```
curl -X POST -d '{"port_name": "MPLS_NETWORK-e1", "type": "linux-htb", "queues": [{"max_rate": "100000000"}]}'  
http://localhost:8080/qos/queue/00000000000000700
```

```
curl -X POST -d '{"port_name": "INTERNET-e1", "type": "linux-htb", "queues": [{"max_rate": "100000000"}]}'  
http://localhost:8080/qos/queue/00000000000000700
```

```
curl -X POST -d '{"port_name": "MPLS_NETWORK-e2", "type": "linux-htb", "queues": [{"max_rate": "100000000"}]}'  
http://localhost:8080/qos/queue/00000000000000800
```

```
curl -X POST -d '{"port_name": "INTERNET-e2", "type": "linux-htb", "queues": [{"max_rate": "100000000"}]}'  
http://localhost:8080/qos/queue/00000000000000800
```

Iperf UDP de 50Mbps

```
> iperf -c 10.1.2.3 -i 1 -u -b 50000000
```

datapath	in-port	eth-src	eth-dst	packets
000000000000700	1	00:00:00:00:01:01	00:00:00:00:05:01	113807
000000000000700	2	00:00:00:00:05:01	00:00:00:00:01:01	83
000000000000800	1	00:00:00:00:03:01	00:00:00:00:05:02	84
000000000000800	2	00:00:00:00:05:02	00:00:00:00:03:01	113808

datapath	port	rx-pkts	tx-pkts	port-bw(Kbps)	port-speed(Kbps)	port-freebw(Kbps)	port-state	link-state
1792	1	114907	113	100000	0.0	100000.0	up	Live
1792	2	111	114907	100000	51429.7	48570.3	up	Live
1792	3	31	29	100000	0.0	100000.0	up	Live
2048	1	112	114908	100000	51427.7	48572.3	up	Live
2048	2	114907	113	100000	0.0	100000.0	up	Live
2048	3	30	30	100000	0.0	100000.0	up	Live

2 Iperf UDP de 50Mbps cada uno

```
> iperf -c 10.1.2.3 -i 1 -P -u -b 50000000
```

datapath	in-port	eth-src	eth-dst	packets
000000000000700	1	00:00:00:00:01:01	00:00:00:00:05:01	278384
000000000000700	2	00:00:00:00:05:01	00:00:00:00:01:01	91
000000000000800	1	00:00:00:00:03:01	00:00:00:00:05:02	92
000000000000800	2	00:00:00:00:05:02	00:00:00:00:03:01	278385

datapath	port	rx-pkts	tx-pkts	port-bw(Kbps)	port-speed(Kbps)	port-freebw(Kbps)	port-state	link-state
1792	1	280706	121	100000	0.0	100000.0	up	Live
1792	2	119	280706	100000	102847.3	0.0	up	Live
1792	3	32	30	100000	0.1	99999.9	up	Live
2048	1	120	280708	100000	102864.6	0.0	up	Live
2048	2	280706	123	100000	0.1	99999.9	up	Live
2048	3	31	31	100000	0.1	99999.9	up	Live

2 Iperf UDP de 40Mbps cada uno

```
> iperf -c 10.1.2.3 -i 1 -P -u -b 40000000
```

datapath	in-port	eth-src	eth-dst	packets
00000000000000700	1	00:00:00:00:01:01	00:00:00:00:05:01	710518
00000000000000700	2	00:00:00:00:05:01	00:00:00:00:01:01	109
00000000000000800	1	00:00:00:00:03:01	00:00:00:00:05:02	109
00000000000000800	2	00:00:00:00:05:02	00:00:00:00:03:01	710518

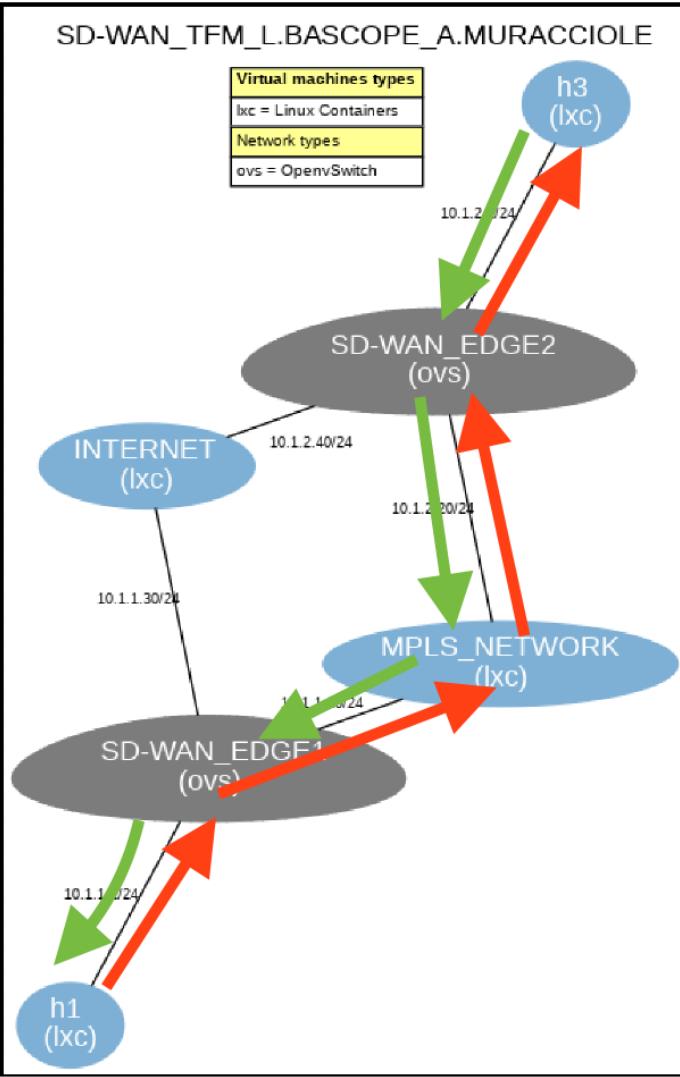
datapath	port	rx-pkts	tx-pkts	port-bw(Kbps)	port-speed(Kbps)	port-freebw(Kbps)	port-state	link-state
1792	1	711066	139	100000	0.0	100000.0	up	Live
1792	2	137	711066	100000	82291.5	17708.5	up	Live
1792	3	32	31	100000	0.0	100000.0	up	Live
2048	1	137	711068	100000	82291.6	17708.4	up	Live
2048	2	711065	140	100000	0.1	99999.9	up	Live
2048	3	31	31	100000	0.0	100000.0	up	Live

PACKET LOSS

- Cálculo de paquetes perdidos en cada enlace

Packet Loss

- No se logró imprimir en pantalla el valor del packet loss pero si se pudo calcularlo cruzando los valores de *flow table* y *port table* obtenidas mediante el Traffic Monitor.
- *Flow table*: Tablas de flujo que va creando el controlador a partir de las direcciones MAC origen y destino
- *Port Table*: Tablas donde se van contabilizando por paquetes emitidos y recibidos de cada puerto de los controladores así como el estado y velocidad de los mismos.



ICMP request

- 1) Paquete ICMP entra por puerto 1 de connm 700
- 2) Paquete ICMP sale por puerto 2 de connm 700
- 3) Paquete ICMP entra por puerto 2 de connm 800
- 4) Paquete ICMP sale por puerto 1 de connm. 800

ICMP reply

- 1) Paquete ICMP entra por puerto 1 de connm 800
- 2) Paquete ICMP sale por puerto 2 de connm 800
- 3) Paquete ICMP entra por puerto 2 de connm 700
- 4) Paquete ICMP sale por puerto 1 de connm. 700

ANTES:

datapath	in-port	eth-src	eth-dst	packets
000000000000700	1	00:00:00:00:01:01	00:00:00:00:05:01	77
000000000000700	2	00:00:00:00:05:01	00:00:00:00:01:01	78
000000000000300	1	00:00:00:00:03:01	00:00:00:00:05:02	78
000000000000300	2	00:00:00:00:05:02	00:00:00:00:03:01	77

+1

datapath	port	rx-pkts	tx-pkts	port-bw(Kbps)	port-speed(Kbps)	port-freebw(Kbps)	port-state	link-stat
1792	1	105	105	100000	0.0	100000.0	up	Live
1792	2	104	105	100000	0.0	100000.0	up	Live
1792	3	29	27	100000	0.0	100000.0	up	Live
2048	1	103	105	100000	0.0	100000.0	up	Live
2048	2	104	105	100000	0.0	100000.0	up	Live
2048	3	29	27	100000	0.0	100000.0	up	Live

+1

DESPUÉS:

datapath	in-port	eth-src	eth-dst	packets
000000000000700	1	00:00:00:00:01:01	00:00:00:00:05:01	78
000000000000700	2	00:00:00:00:05:01	00:00:00:00:01:01	79
000000000000300	1	00:00:00:00:03:01	00:00:00:00:05:02	79
000000000000300	2	00:00:00:00:05:02	00:00:00:00:03:01	78

+1

datapath	port	rx-pkts	tx-pkts	port-bw(Kbps)	port-speed(Kbps)	port-freebw(Kbps)	port-state	link-stat
1792	1	106	106	100000	0.1	99999.9	up	Live
1792	2	105	106	100000	0.1	99999.9	up	Live
1792	3	29	27	100000	0.0	100000.0	up	Live
2048	1	104	106	100000	0.1	99999.9	up	Live
2048	2	105	106	100000	0.1	99999.9	up	Live
2048	3	29	27	100000	0.0	100000.0	up	Live

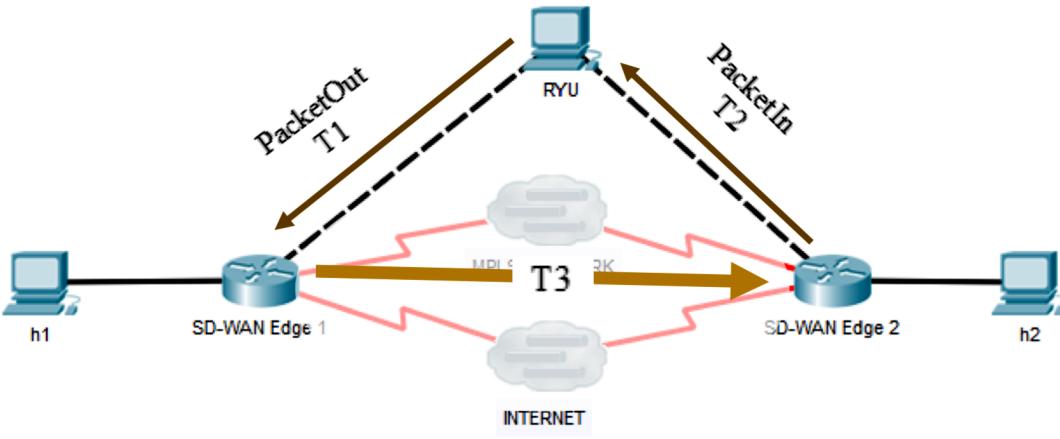
+1

DELAY

- Cálculo de la latencia:

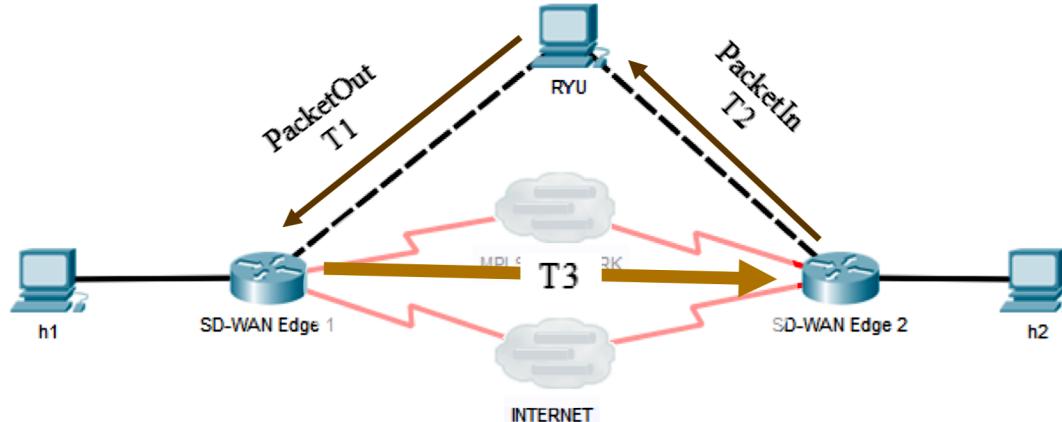
“el tiempo que tarda en trasmitirse un paquete por la red desde un origen hasta su destino”

Latencia



$$T_{total} = T1 + T2 + T3$$

$$T3 = T_{total} - T1 - T2$$



- $T1$: Tiempo que demora el mensaje PacketOut desde el controlador hasta el Switch
- $T2$: Tiempo que demora el mensaje PacketIn desde Switch hasta el controlador

Cálculos de T1 y T2

$$T1 = 0.5 * (Tb - Ta)$$

- ✓ Ta: instante en que se envía el paquete de prueba. En este caso es una solicitud OpenFlow de tipo “ports-stats-request”.
- ✓ Tb: instante en que se recibe la respuesta al mensaje que es de tipo “port-stats-received”.

Planificación de trabajo

- **Periodo I:**
- **Fechas:** 6 de febrero – 14 de febrero
- **Objetivos:**
 - Planteamiento del problema
 - Diseño del Escenario
- **Periodo II:**
- **Fechas:** 17 de febrero – 6 de marzo
- **Objetivos:**
 - Estudio de controlador RyU y sus características

Planificación de trabajo

- **Periodo III:**
- **Fechas:** 7 de marzo – 20 de abril
- **Objetivos:**
 - Estudio de los parámetros de calidad de servicio
 - Presentación final del proyecto

Febrero

FEBRERO															
LU	TAREA	HRS	MA	TAREA	HRS	MIE	TAREA	HRS	JU	TAREA	HRS	VIE	TAREA	HRS	
						5	Elección del Proyecto	2	6	Planteamiento del problema y definición de Objetivos	3	7			
10			11			12	Estudio y prueba de escenarios en un entorno real de enrutamiento, utilizando OVS como Router	4	13	Reunión de trabajo y preparación de presentación	2	14	PRIMERA PRESENTACION DEL PROYECTO	2	
17	DISEÑO DEL ESCENARIO INICIAL	4	18	DISEÑO DEL ESCENARIO DEFINITIVO	4	19	Definición de los parámetros QoS	3	20	Reunión con Tutor	0.5	21	Estudio y pruebas del controlador Ryu	2.5	
24	Modificación de script Traffic Monitor para obtención de parámetros de QoS	4	25	Modificación de script Traffic Monitor para obtención de parámetros de QoS	4	26	Reunión de Trabajo	1	27	Modificación de script Traffic Monitor para obtención de parámetros de QoS	4	28			

Marzo

MARZO															
LU	TAREA	Hrs	MAR	TAREA	Hrs	MIE	TAREA	Hrs	JU	TAREA	Hrs	VIE	TAREA	Hrs	
2	Modificación de script Traffic Monitor para obtención de parámetros de QoS	4	3	Definición de Traffic Monitor como Controlador	3.5	4	Reunión de trabajo y preparación de presentación	3	5	Modificación de script Traffic Monitor para obtención de parámetros de QoS	4	6	SEGUNDA PRESENTACION DEL PROYECTO	2	
9	Pruebas de Traffic Monitor	3	10	Pruebas de Traffic Monitor	1.5	11	Modificación de script Traffic Monitor para obtención de parámetros de QoS	4	12			13			
16	INICIO DEL PERIODO DE CAURENTENA: DEFINICION DE UN NUEVO PLAN DE TRABAJO POR MEDIO DE LA HERRAMIENTA TEAMS		17	Definición del Cálculo y modelado del BW	4	18	REUNION DE ASIGNACION DE PARAMETROS Y DELEGACION DE TAREAS	3	19			20	Definición del Calculo y modelado del BW	6	
23			24		25			26		REUNION DE TRABAJO VIA TELEMATICA	3	27			
30	ESTUDIO DE LA DETERMINACION DE LA LATENCIA	6	31	ESTUDIO DE LA DETERMINACION DE LA LATENCIA	6										

Abril

ABRIL														
LU	TAREA	Hrs	MA	TAREA	Hrs	MIE	TAREA	Hrs	JU	TAREA	Hrs	VIE	TAREA	Hrs
						1			2	ESTUDIO DE LA DETERMINACION DE LA PERDIDA DE PAQUETES	6	3	ESTUDIO DE LA DETERMINACION DE LA PERDIDA DE PAQUETES	6
6			7			8			9			10	REUNION DE PREPARACION PARA LA PRESENTACION FINAL	2
13	REUNION DE PREPARACION PARA LA PRESENTACION FINAL	2	14	REDACCION DE LA MEMORIA	8	15	REDACCION FINAL DE LA MEMORIA	6	16	REUNION FINAL CON EL TUTOR	2	17	ELABORACION DE LA PRESENTACION FINAL	3
20	PRESENTACION FINAL DEL PROYECTO													

Conclusiones

Se realizó un escenario correcto para implementar y obtener los parámetros que se deseaban.

Se hizo un estudio correcto de todas las posibles aplicaciones de Ryu, llegando a determinar la que mas se acercaba a nuestras necesidades.

Se hicieron adaptaciones del Traffic Monitor de forma que se pudo obtener 2 de los 3 parámetros deseados.

Se estudió de forma teórica como debería ser el calculo del Delay

Conclusiones

En una primer instancia la idea era poder plasmar de forma mas visible estos tres parámetros pero no fue posible debido a complicaciones n cuanto a programación y funcionalidades del controlador deseado.

Se propone como trabajo a futuro poder determinar el únicamente el delay a partir de la teoría explicada en dicho proyecto.

Se cumplieron con los tiempos y cronogramas estipulados en un principio. La herramienta Trello fue importante para llevar un orden y seguimiento de las tareas.

Gracias!

Luis Leonardo Bascopé Aparicio

Andrés Jorge Muracciole Vázquez

Miniproyectos – 2019/2020



POLITÉCNICA

UNIVERSIDAD
POLÍTÉCNICA
DE MADRID