PUCE

Maestría en Redes de Comunicaciones

Tecnologías en Redes de Banda Ancha

Trabajo Práctico:
Medición del Jitter

Integrantes:

Luis Aguas *Paul Bernal*
David Badillo *Ernesto Pérez*

Diciembre 2012

Índice de Contenidos

Objetivo:

- Definir un escenario en el cual dos grupos de máquinas se conecten a través de un equipo que les controle el ancho de banda. Simulando una red WAN. En este caso definimos usar un canal de 1mbps de ancho de banda entre ambos grupos.
- Aplicar QoS a través de :
    - Lograr marcar paquetes según su contenido, puerto de destino, tipo de protocolo, etc.
    - Una vez marcados, asignarles a colas con prioridad (alta, media, baja)
- Saturar el ancho de banda y medir el jitter en una comunicación telefónica (VoIP) en el escenario de que tengamos QoS activado/desactivado

Variantes analizadas

En principio nos orientamos a utilizar el sistema de htb y el de cbq para clasificar los paquetes. Sin embargo el uso de estos scripts es bastante crítico, incluso para los que conocemos de Linux.

Por tanto buscamos una solución que nos permitiera realizar la misma labor, pues a la final encontramos un sistema llamado ZeroShell que está basado en Linux, se configura a través de la web y utiliza las mismas herramientas para clasificar y encolar tráfico: tc, iptables

Nuestro primer aporte fue describir en el foro de la asignatura una pequeña descripción paso a paso de cómo echar a andar el escenario para que el ZeroShell pudiera actuar como el sistema que controla el ancho de banda y encola paquetes.

Infraestructura:

Tenemos 5 equipos virtuales, el primero, llamado “debian” es el punto de acceso a la red virtual, es decir, este es el único equipo que tiene conectividad con el mundo real y por ello es a través del cual podremos conectarnos a todo el entorno del experimento remotamente.

El esquema de red tiene un equipo en el centro, llamado “ZS” (de ZeroShell) que está haciendo las funciones de bridging entre las redes que llamaremos: de la izquierda y de la derecha, porque en el esquema lógico estas ocupan dichas disposiciones.

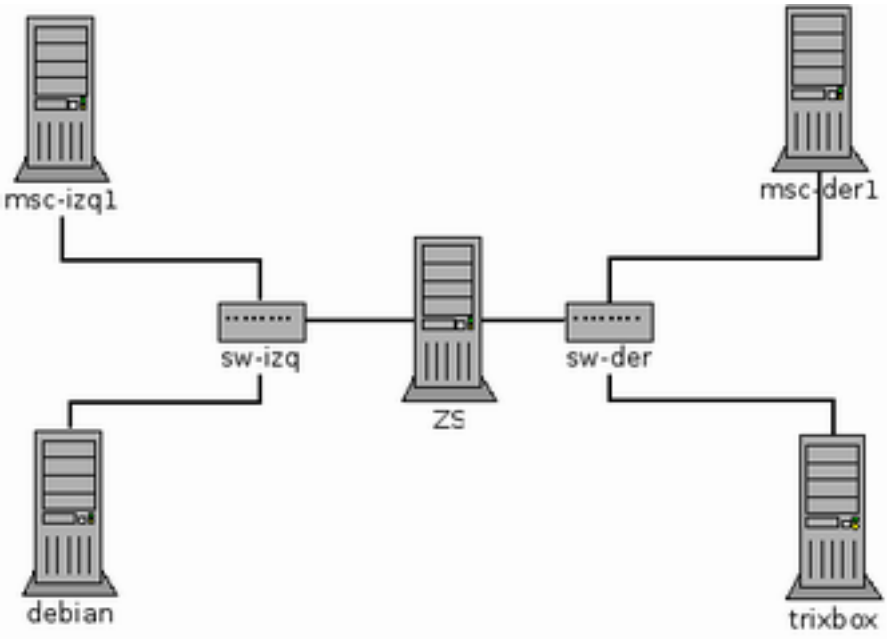
En la red de la izquierda están el equipo “debian” y el llamado “msc-izq1” con CentOS, ambos conectados a un switch virtual al cual también está conectada la tarjeta de la izquierda del “ZS” que podría ser ETH00 dentro del ZS.

En la red de la derecha está un equipo llamado “msc-der1” también con CentOS, junto con un “trixbox” que es nuestra central telefónica. Igualmente ambos conectados a un switch virtual al cual también está enlazada la tarjeta de la derecha del “ZS” que podría ser ETH01 dentro del ZS.

Localmente los equipos son accesibles a través reglas de NAT aplicadas en el “debian”, de la siguiente forma:

Nombre	IP Remota	IP Local	IP LAN virtual	Puerto en WAN	Puerto en LAN
debian	188.69.44.221	192.168.**9**.243	192.168.**1**.1	2222	22
ZS	188.69.44.221	192.168.**9**.243	192.168.**1**.230	80, 443	80, 443
msc-izq1	188.69.44.221	192.168.**9**.243	192.168.**1**.241	241	241
msc-der1	188.69.44.221	192.168.**9**.243	192.168.**1**.251	251	251
trixbox	188.69.44.221	192.168.**9**.243	192.168.**1**.252	252	252

Dado que todos los equipos del experimento (excepto “debian”) están únicamente conectados a un entorno de red totalmente virtual, todos ellos son accesibles desde fuera a través de las IP Remota y Local como mostramos en la anterior tabla.



Topología lógica de la red

Configuración del ZeroShell

ZeroShell está configurado para hacer traffic shapping de 1mbit/s.

En ETH00

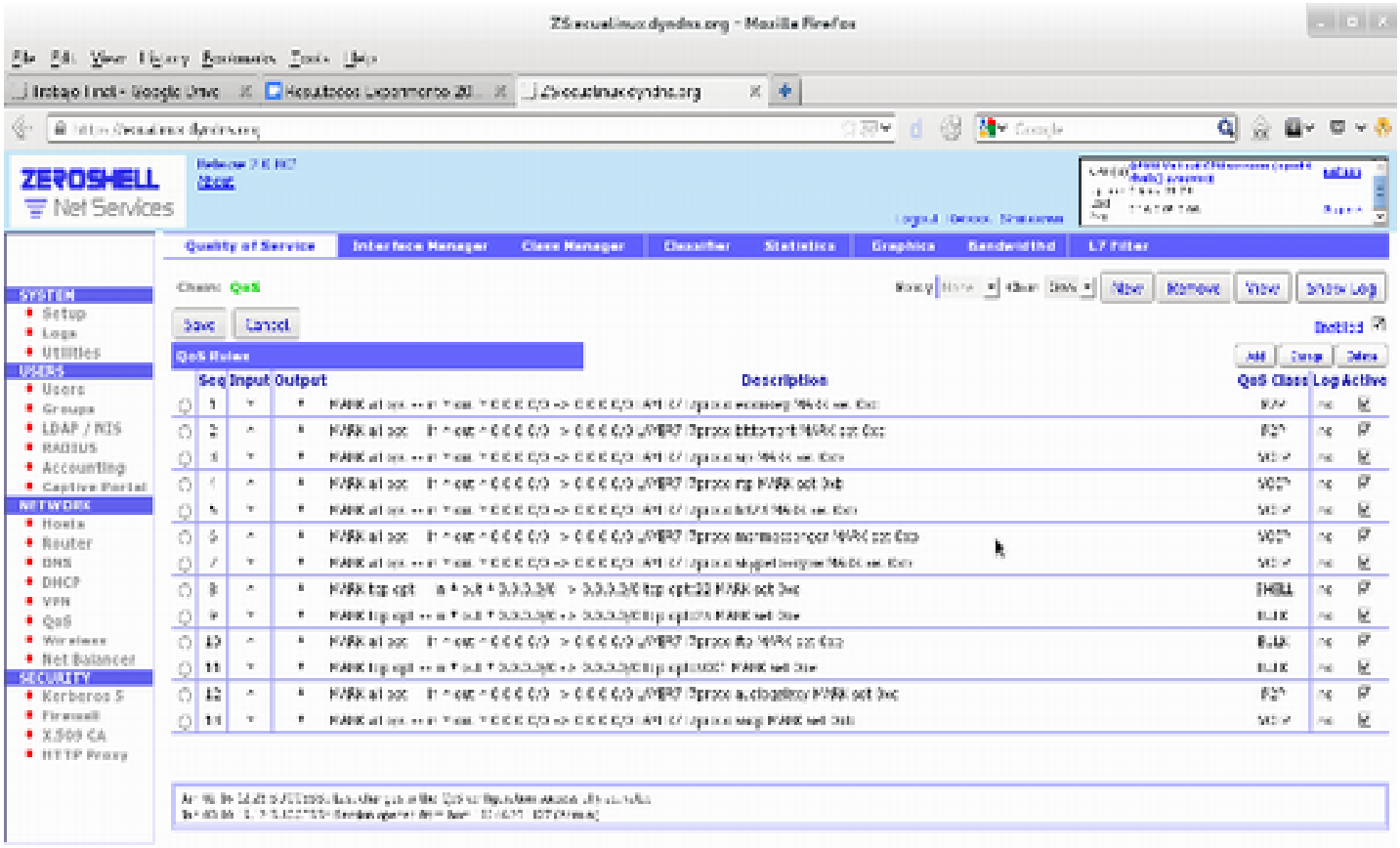
En ETH01

Se siguió el manual de ZS para bridging y se clasificaron los paquetes:

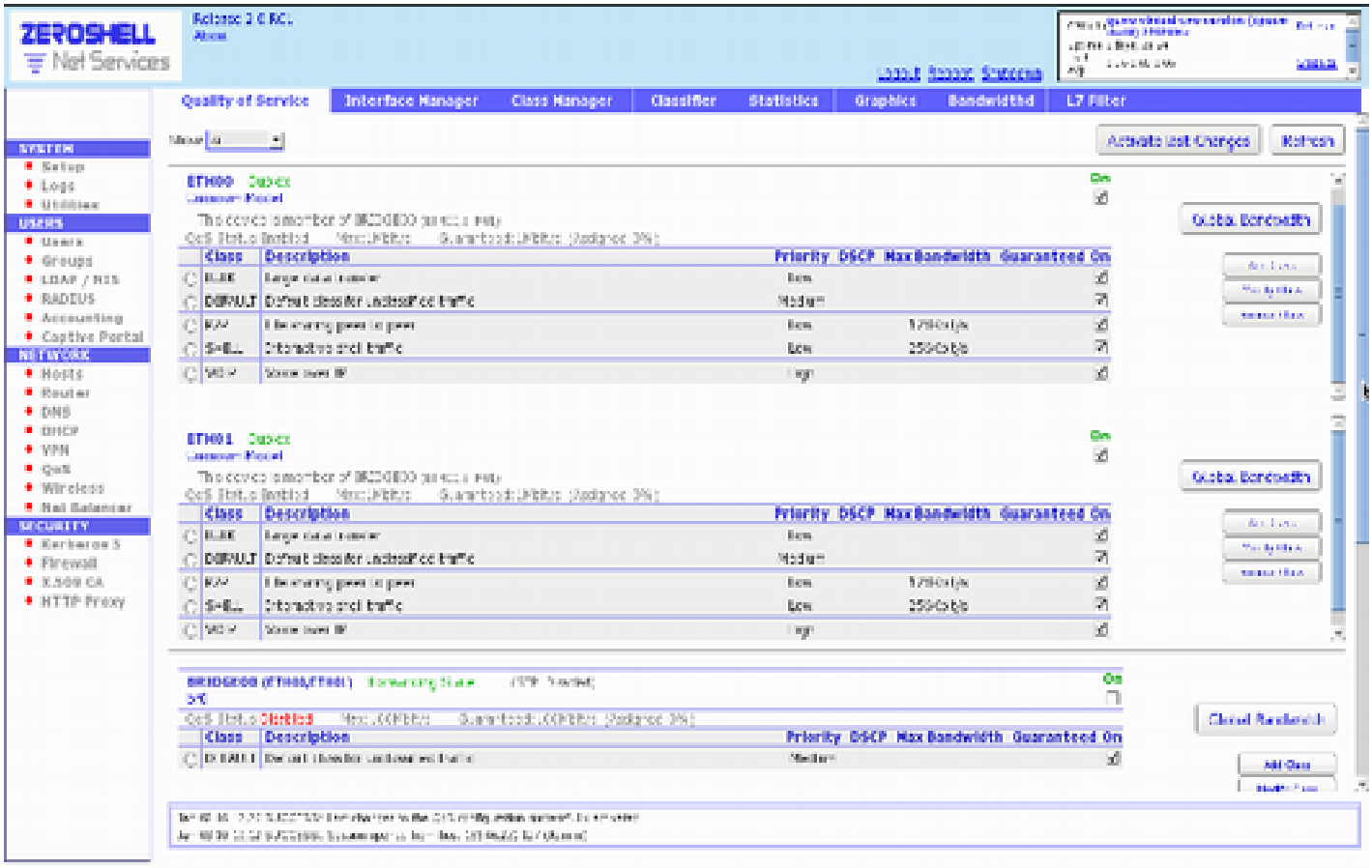
- Prioridad Alta: VoIP (RTP, SIP, Skype, etc)
- Prioridad Media: ssh y otros protocolos
- Prioridad Baja: puerto 5001 que es el que Iperf usa para la transferencia

En resumen, VoIP tendrá la máxima prioridad, sobre cualquier otro protocolo.

Clasificación:



Para cada cola, se definieron prioridades en ambas tarjetas:



Llamadas e Iperf

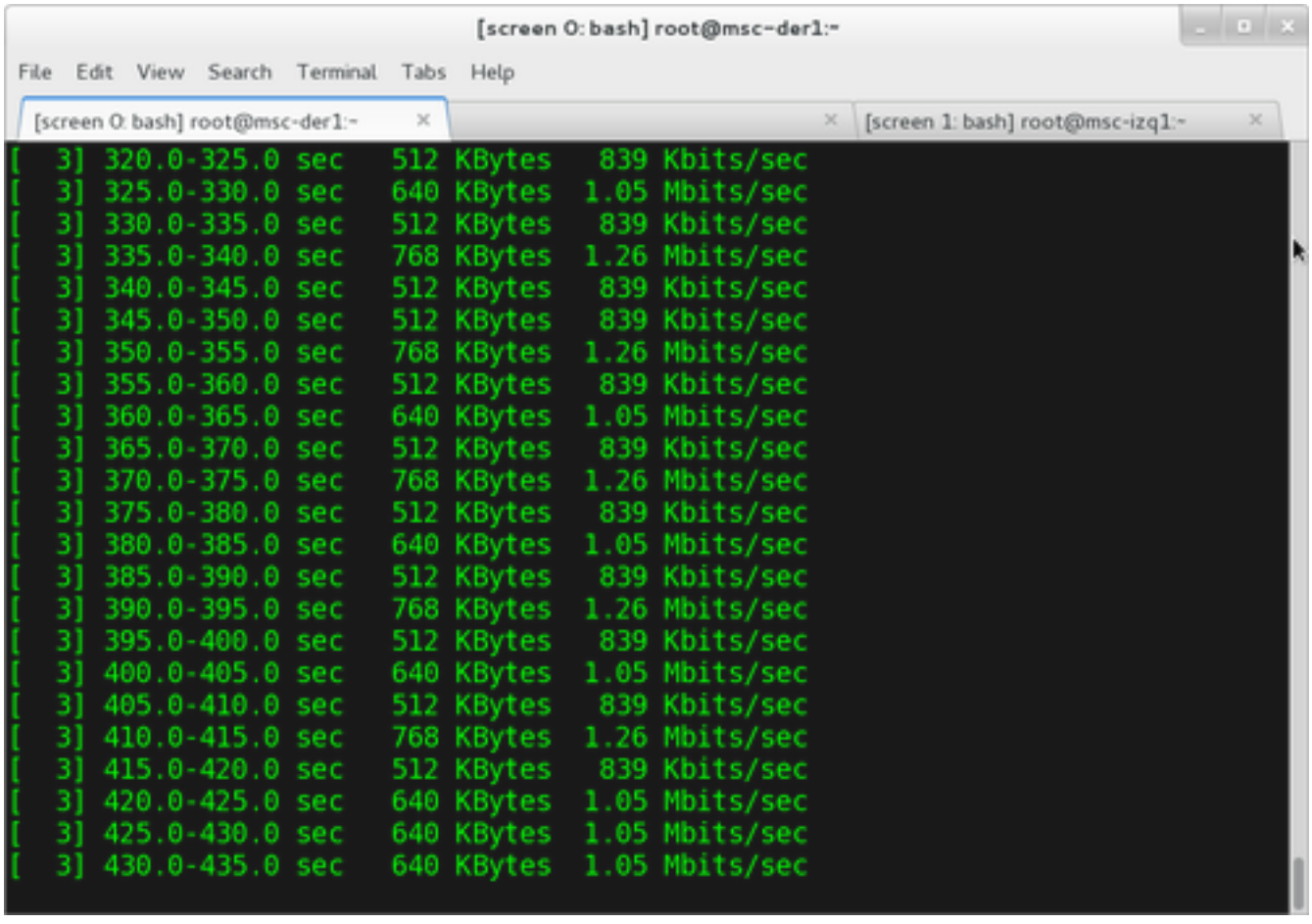
Las llamadas telefónicas se realizaron desde msc-izq1 hacia el trixbox usando un softphone (usamos twinkle y ekiga). Para los muestreos se usaron llamadas de un promedio de tiempo de 20 segundos.

La saturación de tráfico se realizó desde “debian” hacia “msc-der1” usando “iperf” con el siguiente comando:

iperf -c 192.168.1.1 -t 3600 -i 5

- -t 3600 = correrle durante 3600 segundos
- -i 5 = reportar consumo cada 5 segundos

Una imagen así se vería del tráfico generado:



Las mediciones de jitter se hicieron usando wireshark corriendo sobre “msc-izq1”.

Sobre este esquema de red se trabajaron 4 escenarios en los cuales variaron dos parámetros:

- Iperf activado/desactivado

- Encolamiento en ZS activado/desactivado. Para activar/desactivar, simplemente fuimos al menú "Interface Manager" (Ver imagen en la página anterior) y marcamos o desmarcamos todas las clases aquí definidas. Excepto la clase DEFAULT para que todo vaya a un sólo tipo de clase.

Trixbox:

Creamos una extensión, la 200, para que el twinkle se conectara a él:
*trixbox1*CLI> sip show peer 200*
*trixbox1*CLI>*

.
.
.
Addr->IP : 192.168.1.241 Port 5060
.
.
.
Useragent : Twinkle/1.4.2
Reg. Contact : sip:200@192.168.1.241
.
.
.

Twinkle:

Le configuramos como a todo softphone,
Imagen de la configuración:

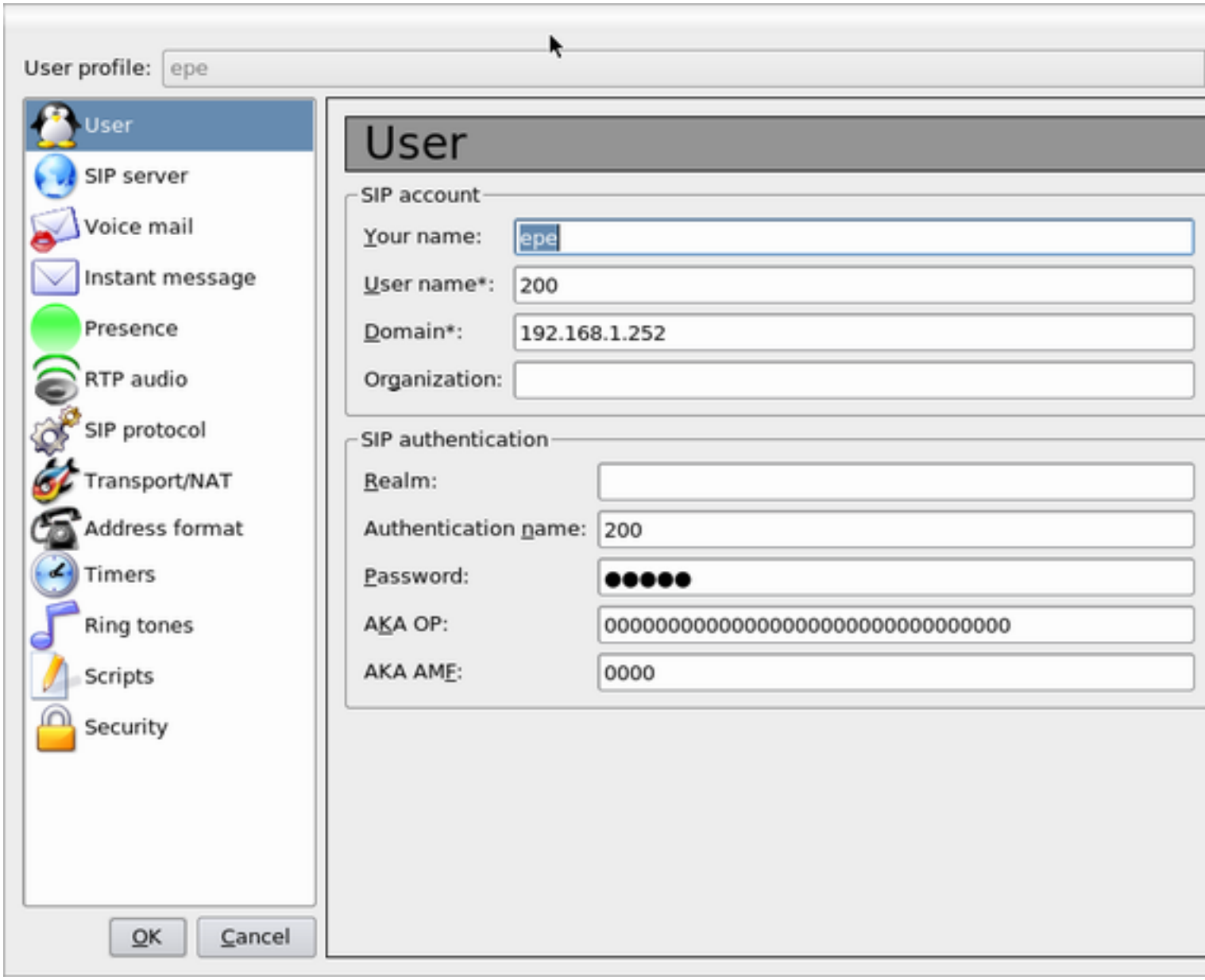
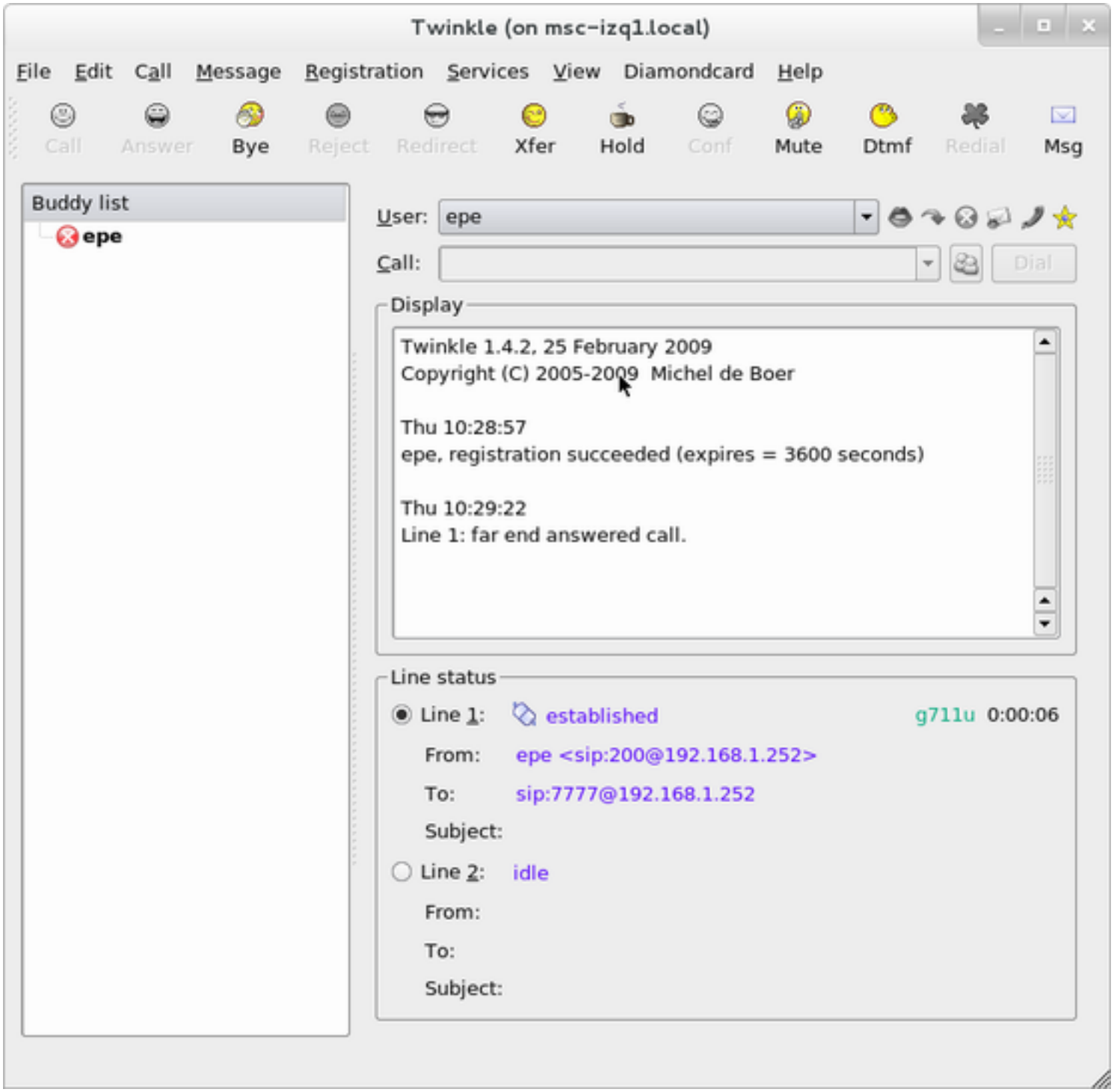


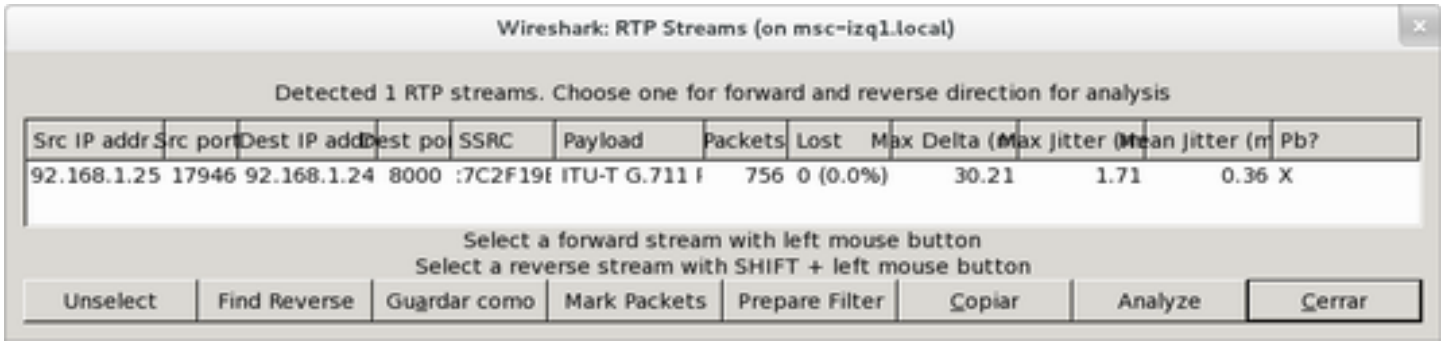
Imagen de una llamada desde twinkle a la extensión 7777 (usamos el codec g711 ulaw):

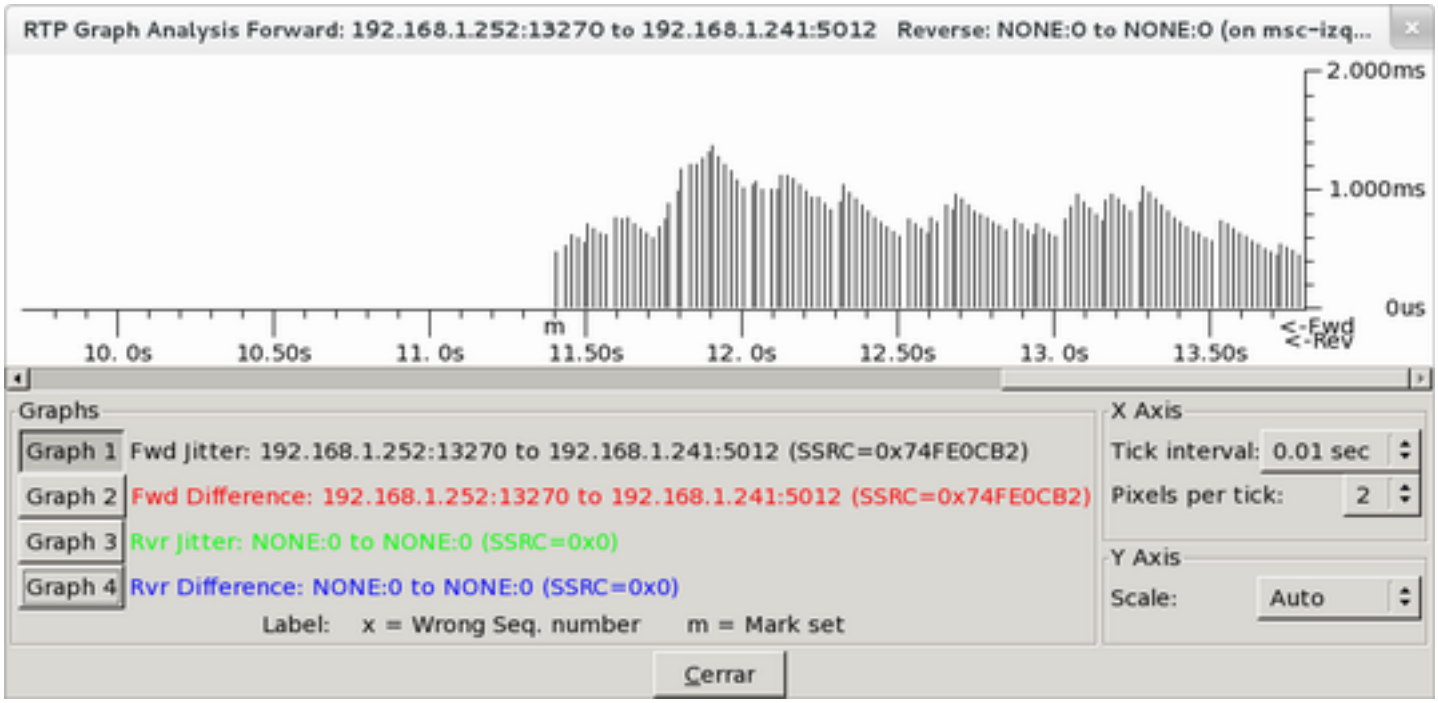


Resultados de las pruebas programadas:

1. “ZS” sin moldeado de tráfico y con el tráfico de red desocupado (iperf desactivado).

Lost Packets	Max Delta	Max Jitter	Mean Jitter
0	30.21	1.71	0.36

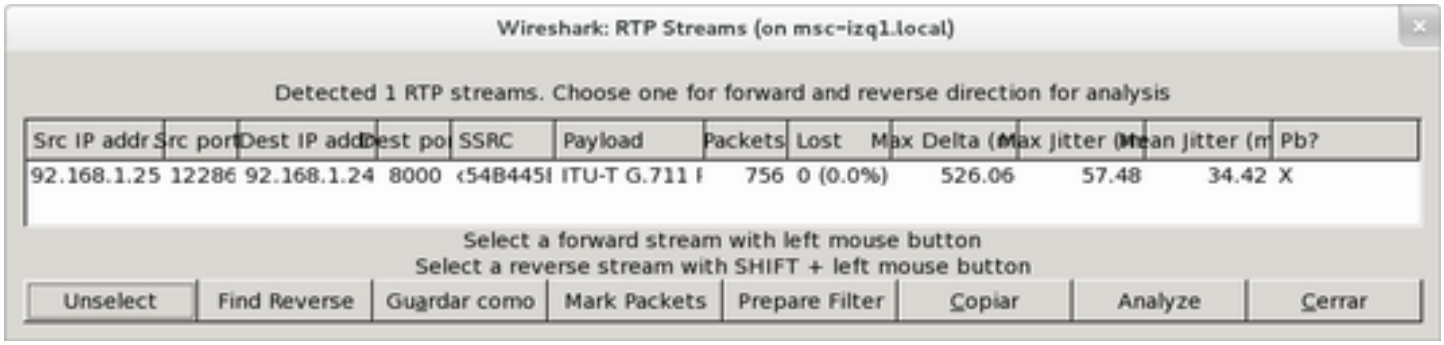


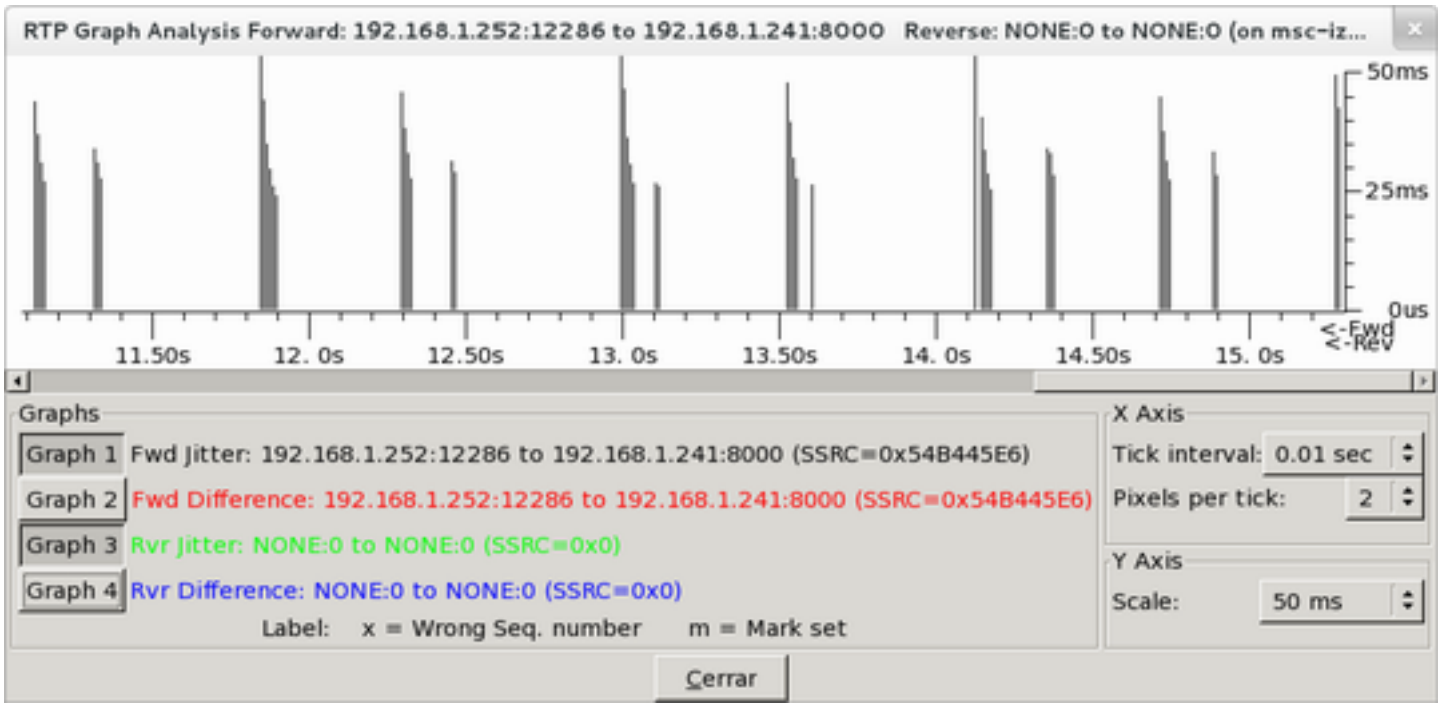


Como era de esperar: al no haber tráfico en la red por parte de iperf, solamente había tráfico para la conexión de VoIP y por tanto las llamadas salieron con un bajo jitter. Notemos la media de 0.36ms y el máximo jitter de 1.71ms

2. “ZS” sin moldeado de tráfico y con el tráfico de red saturado (iperf activo).

Lost Packets	Max Delta	Max Jitter	Mean Jitter
0	526.06	57.48	34.42

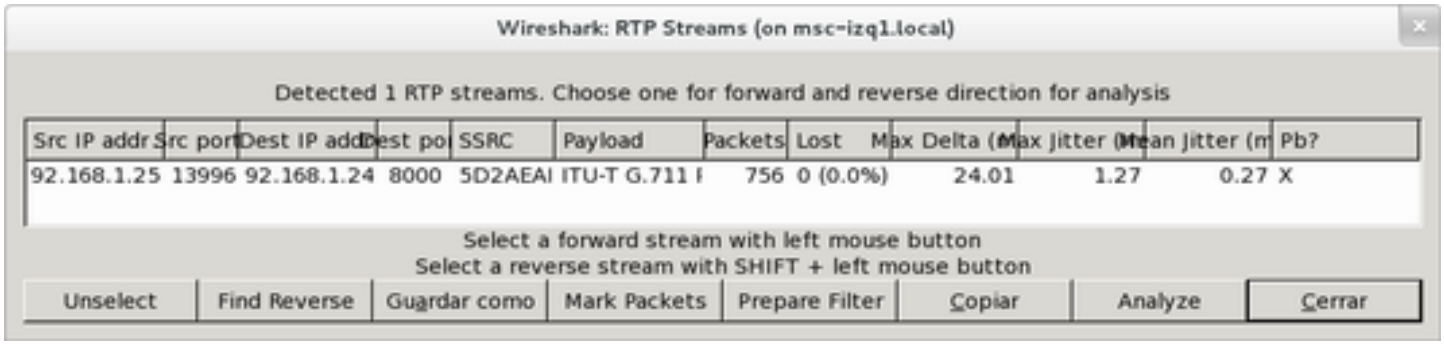


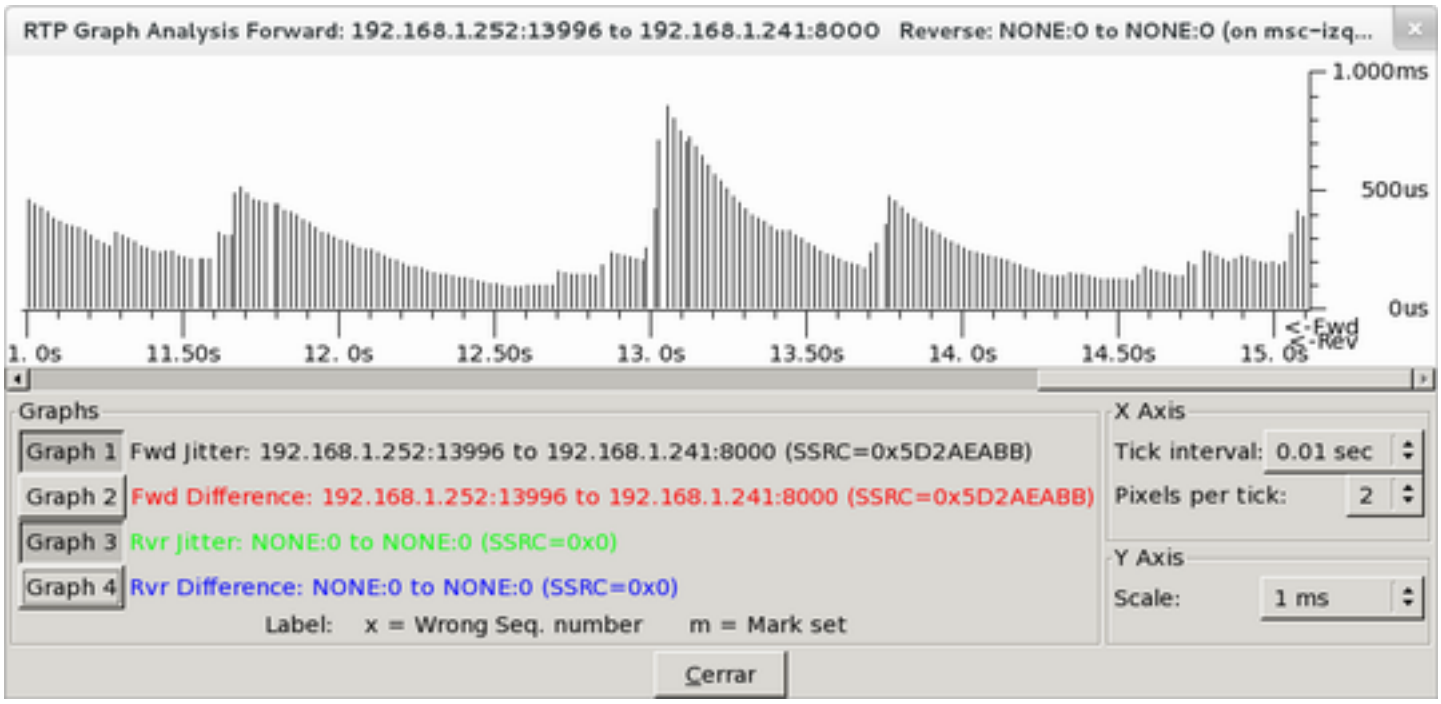


En este escenario, sin encolamiento, y con iperf consumiendo todo el tráfico de red, notamos cómo la llamada telefónica presenta un apreciable jitter (muy alto), vemos una media de 34.42ms y el jitter máximo de 57.48ms. Bajo estas condiciones de saturación del canal de 1mbps simple, sin aplicar ninguna técnica de QoS podemos ver que se degrada el servicio de VoIP

3. “ZS” con moldeado de tráfico y con el tráfico de red desocupado (iperf inactivo).

Lost Packets	Max Delta	Max Jitter	Mean Jitter
0	24.01	1.27	0.27

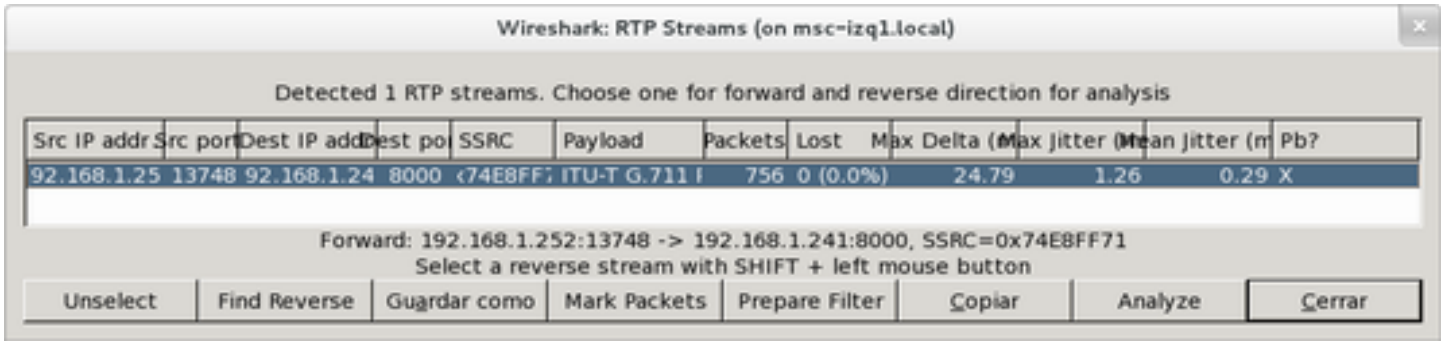


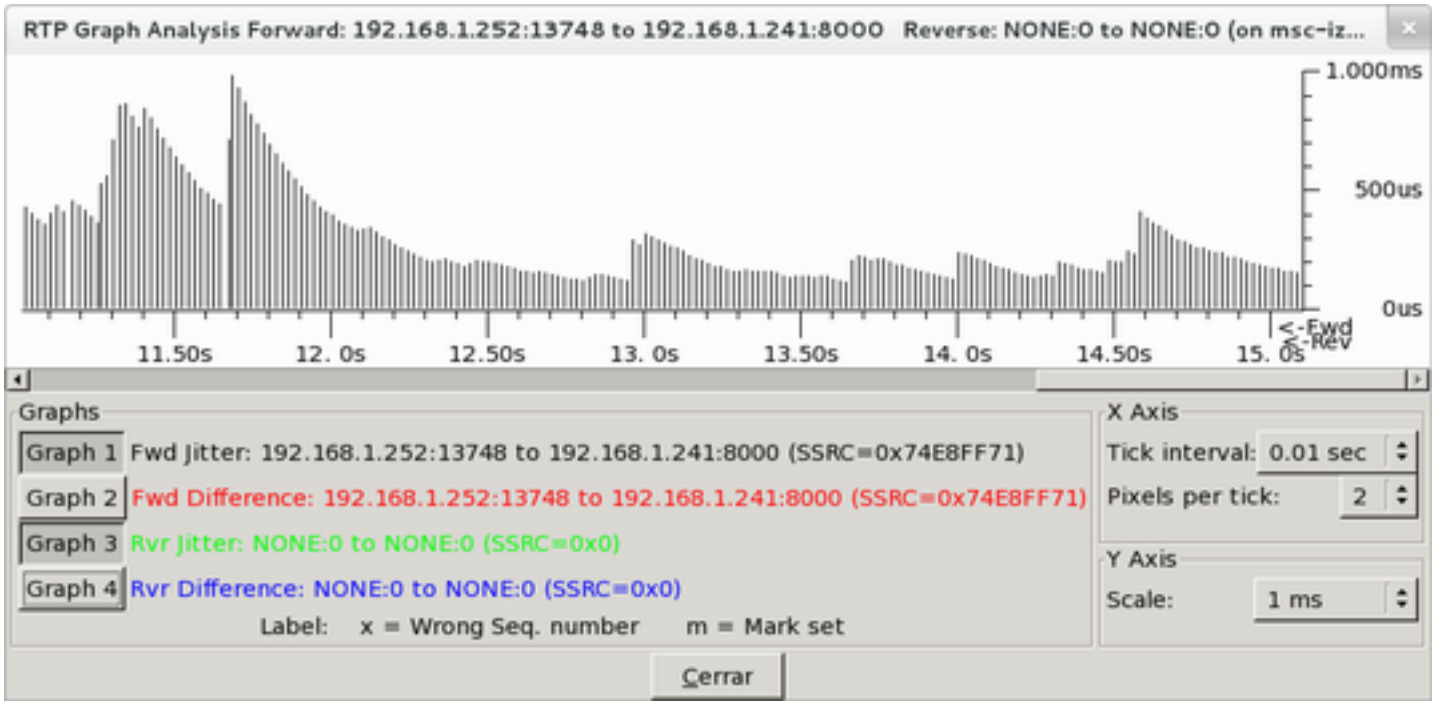


Este escenario lo hicimos solamente con finalidad de mantener el experimento consistente para evaluar los 2 parámetros que estábamos probando, pues suponíamos desde un inicio el resultado de este escenario: iperf no está corriendo, por tanto no hay congestionamiento del canal; pero sí ya aplicamos las técnicas de QoS encolando paquetes con prioridad: máxima prioridad para VoIP y mínima para el tráfico de IPerf (5001). En este caso, el jitter lógicamente fue bajo debido a que no había saturación por tanto nos daba igual con o sin QoS funcionando.media de 0.27ms y máximo de 1.27ms.

4. “ZS” con moldeado de tráfico y con el tráfico de red saturado (iperf activo).

Lost Packets	Max Delta	Max Jitter	Mean Jitter
0	24.79	1.26	0.29





Este escenario es el más interesante pues nos permite ver la funcionalidad del sistema de QoS mediante encolamiento. Tenemos el iperf corriendo y por tanto tratando de consumir todo el canal, es parecido al escenario 2. Pero en este caso tenemos el sistema de QoS funcionando, priorizando los paquetes de VoIP y en una cola con baja prioridad al tráfico al puerto tcp/5001 (iperf). Lógicamente se podía haber agregado en esta cola de baja prioridad a cualquier otro tráfico o al tráfico adicional que desee seleccionar. En base a este escenario las mediciones nos indican que el jitter medio fue de 0.29ms y un jitter máximo de 1.26. Tal y como se comportaría la llamada de VoIP en un escenario en el cual no había congestionamiento.

Resumen:

El control de tráfico a través de aplicar disciplina de encolamiento previo a la detección y clasificación de paquetes en colas, sí funciona.
Es altamente recomendado, en caso de usar VoIP en la red, aplicar técnicas de QoS mediante clasificación y encolamiento de paquetes en colas con prioridad.