QoS en GNU/Linux: su aplicación sobre infraestructura para internet libre

Javier Charne[1], Diego De La Riva[2], Hugo Ramón[3], Adrián Jaszczyszyn[4]

{javier,delariva,hugoramon,adrianjasz}@unnoba.edu.ar
Instituto de Investigación y Transferencia en Tecnología – IITT
Escuela de Tecnología – UNNOBA
Calle Jorge Newbery y Sarmiento – (6000) Junin, Bs As., Argentina

Abstract. Con la aparición del transporte de VoIP (Voice over IP) y los equipos de Videoconferencia/Telepresencia a relativo bajo costo, las empresas comenzaron a utilizar un mismo proveedor para datos, telefonía y video, haciendo converger tres tipos de tráfico -con características y requerimiento distintos- en un mismo enlace físico. Las actuales redes convergentes transportan paquetes que pertenecen a diferentes tipos de servicios (datos, voz, video) y uno de los criterios habituales para clasificarlos distingue el tráfico que tiene requisitos temporales estrictos del resto de los servicios. El ancho de banda siempre limitado de los enlaces, es preciso identificar y privilegiar el procesamiento de este algún tipo de tráfico sobre otros que comparten el medio. En este trabajo se presenta la evaluación y aplicación de diferentes herramientas de QoS (Quality of Service) sobre infraestructura de internet libre en espacios públicos de la ciudad de Junín.

Keywords: VoIP, Videoconferencia, Calidad de Servicio, Tiempo Real, Enlaces, Internet Libre, Linux

1 Introducción

Antes de que comenzara a aparecer el término “redes convergentes” en la bibliografía especializada, y de que éstas fueran ofrecidas por los distintos proveedores, las empresas utilizaban distintos medios para vehiculizar cada tipo de tráfico. La telefonía era transportada a través de la PSTN (Public Switched Telephone Network) mientras que se contrataba un servicio aparte para el transporte de datos, típicamente un enlace Frame-Relay.

[1] Profesor Adjunto con Semidedicación, Escuela de Tecnología, UNNOBA.

[2] Profesor Adjunto con Semidedicación, Escuela de Tecnología, UNNOBA.

[3] Profesor Titular con Dedicación Exclusiva, Escuela de Tecnología, UNNOBA

[4]JTP con Semidedicación, Escuela de Tecnología, UNNOBA

El servicio de video requería un servicio satelital de alto costo. Con la aparición del transporte de VoIP y los equipos de Telepresencia a bajo costo, se comenzó a utilizar un mismo proveedor, tanto para datos, telefonía y video, haciendo converger tres tipos de tráfico con características distintas en un mismo enlace físico.

En [1] se clasifica el tráfico en base a dos grandes categorías: tráfico elástico y no-elástico.

El tráfico elástico se puede ajustar, a los cambios de retardos y rendimientos de una red, sin dejar de satisfacer las necesidades de sus aplicaciones. Éste es el tipo de tráfico soportado por las redes basadas en TCP/IP, (Transmission Control Protocol/Internet Protocol), el tráfico de conexiones individuales se adapta a la congestión reduciendo la velocidad de entrega de datos a la red.

Las aplicaciones de internet, como la transferencia de archivos, el correo electrónico, las conexiones remotas, la gestión de la red, el acceso a la web son con tráfico elástico, aunque hay diferencias entre los requisitos de esas aplicaciones.

Con tráfico elástico, es beneficioso disponer de un servicio de red que distinguiera entre uno y otro tipo de flujo (con QoS). Sin esto, los routers gestionan a ciegas los paquetes IP, sin importarles el tipo de aplicaciones que lo produjo o que está esperándolo, ni si un paquete forma parte de una trasferencia grande o pequeña. Si se produce la congestión, es improbable que se asignen los recursos de modo que se satisfagan las necesidades de todas las aplicaciones. Cuando además se incorpora el tráfico no-elástico, la experiencia es menos satisfactoria.

El tráfico no-elástico no se adapta a las variaciones de retardos y rendimiento de redes interconectadas. El principal ejemplo, es el tráfico de tiempo real, como voz y video. Los requisitos del tráfico no-elástico pueden enumerarse como: requiere un mínimo valor de ***rendimiento***. Muchas aplicaciones no-elásticas exigen un mínimo de rendimiento consistente. Los paquetes no se deben ***acelerar***, ***retrasar***. Las aplicaciones de tiempo real difieren en la cantidad de ***paquetes perdidos*** que admiten.

Estos son difíciles de cumplir en una red congestionada, con retardos variables en colas y pérdidas de paquetes. El tráfico elástico introduce dos requisitos en la arquitectura de interconexión de redes. En primer lugar se necesita algún mecanismo para otorgar un tratamiento preferente a las aplicaciones que tengan requisitos más exigentes, ya sea por adelantado con algún tipo de función de solicitud de servicio, o bien on-the-fly, usando los campos de la cabecera de los paquetes IP y permite a la red anticiparse a la demanda y denegar nuevas solicitudes si los recursos solicitados no están disponibles y podría darse el caso de una congestión. Este enfoque implica el uso de algún tipo de protocolo de reserva de recursos.

Otro requisito para el soporte del tráfico no-elástico en una arquitectura de internets es que el tráfico elástico debería contemplar el tráfico no-elástico. TCP realiza un control de flujo en presencia de la congestión, UDP no lo hace, y no reduce su demanda, ni siquiera detecta la situación. Se debe implementar un protocolo de reserva de recursos que controle la situación, denegando nuevas solicitudes de transmisión si no queda ancho de banda disponible para el tráfico elástico.

2 Mecanismos de QoS

Los distintos mecanismos de QoS son los que, en conjunto, hacen posible implementarla dentro de la red IP y deben llevarse a cabo en un cierto orden como si fuera una cadena de producción industrial: Clasificación y Marcado, Policing & Shaping y Administración de la congestión.

Todos los equipos de comunicación poseen buffers destinados a guardar con formato de trama los datos que ingresan por una interfaz. Estos buffers suelen ser memorias independientes, uno por cada interfaz. Y como el tiempo de procesamiento de las tramas que llegan puede ser mayor que la tasa de entrada, se almacenan en esta memoria para luego ser procesadas.

Para las tramas salientes, la utilidad de estos buffers es diferente, pues en ellos van a residir las tramas que están "listas para salir" por la red. También serán memorias físicamente independiente pero su asignación a cada interfaz no es uno-a-uno, sino que cada interfaz física puede tener asignadas varias de estas memorias, denominadas colas.

La *clasificación* del tráfico es simplemente la identificación de cierto tipo de flujo. La mayoría de los IOS (Internetwork Operating System) [2] que implementan mecanismos de QoS permiten clasificar mediante las marcas en los campos CoS (Class of Service) y ToS (Type of Service). Luego la acción que no afecta al tráfico es el *marcado* que puede hacerse en alguna estructura del sistema operativo o en el campo CoS o ToS.

Para gestionar el ancho de banda luego de tener un paquete marcado la decisión pasa por si se envía, se retrasa o se descarta. *Policing* no tiene en cuenta el comportamiento del tráfico mientras que *shaping* si lo hace [3].

Administración de la congestión: una vez que los paquetes fueron clasificados y marcados, y que se estableció qué se debe hacer con ellos, resta procesarlos y -si el tráfico es saliente- despacharlos por la interfaz elegida, teniendo especial cuidado aquí de no congestionar el enlace. Un enlace saturado es aquel que llega al límite físico de transmisión, o bien de la memoria asignada a dicha interfaz.

Los paquetes despachados se toman de las colas de salida, y el criterio para seleccionar con que cola comenzar, se denomina qdiscs (queueing disciplines). Es importante que el proceso de QoS se realice en el dispositivo que realmente posee la cola real.

3 Junin I/L

El proyecto Junín-IL (Intenet Libre) surgió como un requerimiento simple: *"Se quiere Internet en las plazas"*. Y en cuando se reunió el equipo de trabajo, surgieron las preguntas:

- ¿Cuáles plazas?
- ¿Qué área debemos abarcar?
- ¿Hacemos nosotros el proyecto o subcontratamos alguna solución "llave en mano"?

- ¿Cómo llegamos con la red hasta las plazas?
- ¿Cableamos por los postes de luz y luego ponemos un AP en cada plaza?
- ¿Ponemos un router que distribuya wifi en algún edificio cercano y cableamos hasta ahí desde el NOC?
- ¿qué solución es más escalable?
- ¿Qué calidad de servicio queremos que el cliente experimente?
- ¿Cuál va a ser el ancho de banda total de salida?
- ¿Cuántos clientes concurrentes esperamos?
- ¿Va a ser navegación liberada o vamos a restringir algún sitio?

Los espacios públicos comprendían: la plaza principal “25 de mayo”, la terminal de ómnibus y sus plazas linderas, y la plaza Ferrocarriles Argentinos (Fig. 1.). La navegación libre, pero debía limitar la velocidad a la que cada usuario puede bajar datos de Internet, privilegiando la navegación web por sobre los demás servicios.

Fig. 1

Se investigaron varias soluciones en producción, como UNLP [16] y San Luis [17].

Se consultaron presupuesto y prestaciones de las soluciones WLC para wireless outdoor Aironet de Cisco [18] [19] [20] y también de equipamiento Wavion [21]. Finalmente, se decide hacer la instalación con equipos Planet Mesh Network MAP-2000 y MAP-2000R [22] tanto para el troncal de la red inalámbrica como para la distribución de internet en los clientes. Sumado a esto, se realizó el tendido de fibra óptica desde el NOC hasta el equipo principal del mesh, y se instaló un server para hacer el trabajo de enrutamiento entre la LAN y el enlace WAN, firewall, proxy, DNS, QoS y monitoreo.

Los equipos inalámbricos se comunican entre sí en la frecuencia de 5.8Mhz,y redistribuyen la señal localmente en los 2.4Ghz. La ubicación de cada uno se hizo en base a la zona que debían irradiar. Estos equipos fueron instalados en torres en cada locación, elevados a un mínimo de 25 metros.

3.1 Detalles de Implementación

La fig. 2 muestra la topología. Dado que la configuración por defecto de los equipos clientes tienen predeterminada una red 10.0.0.0/8 en su interfaz 802.11a, se optó por dejar este rango en los enlaces contra el equipo central, para que facilite la instalación y administración inicial de futuras ampliaciones.

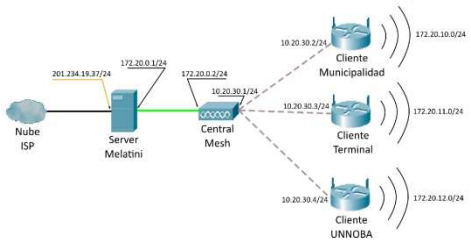

Fig. 2

Cada equipo cliente (Municipalidad, Terminal de Ómnibus y Unnoba) enruta el tráfico de los clientes hacia el equipo central, y tiene configurado un servidor DHCP que distribuye una subred de 172.20.0.0/16 entre sus clientes. Se deshabilitaron las opciones de NAT en cada equipo cliente, buscando que la IP del usuario final llegue al servidor, donde se especifican las políticas de QoS.

El servidor se instaló con el sistema operativo GNU/Linux Debian 6.0 “Squeeze” en el que se instalaron los siguientes paquetes, para proveer los servicios requeridos:

- Servidor de Nombres de Dominio, DNS: **bind9**, versión 1:9.7.2.dfsg.P3-1.1
- Secure Shell, SSH: **openssh-server**, versión 1:5.5p1-6
- Proxy web: **squid3**, versión 3.1.6-1.2
- Servidor web: **apache2**, versión 2.2.16-6
- Servidor de Tiempo: **ntp**, versión 1:4.2.6.p2+dfsg-1+b1
- Servicio de SNMP: **snmpd,** versión 5.4.3~dfsg-2
- Servidor VPN: **openvpn,** versión 2.1.3-2

Además: **munin,** para monitorear el estado del servidor y **sarg**, para generar reportes de uso del proxy Las mayoría de las distribuciones GNU/Linux actuales tienen en sus repositorios una amplia gama de herramientas para la administración, control y monitoreo del tráfico de red [4]. El presente trabajo se basa en la distribución GNU/Linux Debian [5], y se van a resaltar algunas herramientas que, utilizadas en conjunto, otorgan al sistema la capacidad de gestionar el tráfico eficientemente y dotar de calidad a los servicios que provee Junin/IL. Las herramientas utilizadas son iptables [6], iproute [7], squid [8] y las de monitoreo para mucho tráfico como snmpd, mrtg [9], RRDTool [10], wireshark [11], iptraf [12], somokeping [13], flowscan [14], icinga [15].

3.2 Configuraciones.

Para lograr brindar la calidad de servicio requerida, y para restringir el consumo de ancho de banda por usuario a 256Kbps, utilizamos los *delay pools* del proxy *Squid.* Con esto se puede establecer colas por cliente (identificado por su IP), o por subred. Luego, la cola es configurada para permitir una ancho de banda máximo. Cuando un

cliente llega a este límite, los paquetes se retrasan para adecuar el tráfico a la tasa especificada.

El tráfico web proveniente de la red wifi se redirecciona para que pase por el proxy:

```
iptables -t nat -A PREROUTING -i ${LAN_WIFI} -p tcp --
dport 80 -j REDIRECT --to-port 3128
```

Hay dos valores importantes: el primero establece la tasa a la que el tráfico va a adecuarse; el segundo, cuál es la tasa inicial por cliente. Dado que la navegación web usualmente se presenta en ráfagas, es bueno que el cliente pueda bajar de entrada una cantidad mayor de datos (1MB), para después limitarse a la tasa indicada. Los límites se especifican en Bytes, por eso para 256Kbps especificamos 32000B

Se declaran dos colas: una ilimitada (para clientes conectados por VPN para realizar tareas de administración) y otra con los límites requeridos. Excede a este trabajo la presentación de los detalles de tipos de delay pools en Squid, comportamientos predeterminados y su configuración. La configuración del proxy está en el archivo /etc/squid3/squid.conf, y se transcribe aquí la parte de los delay pools:

```
acl lan_wifi src 172.20.0.0/16
acl vpn src 192.168.138.0/24
…
delay_pools 2
delay_initial_bucket_level 50
delay_class 1 1
delay_parameters 1 -1/-1
delay_class 2 3
delay_parameters 2 -1/-1 -1/-1 32000/128000
delay_access 1 allow vpn
delay_access 1 deny all
delay_access 2 allow lan_wifi
delay_access 2 deny all
```

Todo tráfico que no pasa por el proxy, se crean colas (qdisc[24]). En el siguiente script INT_IF es la interfaz interna y EXT_IF es la interfaz conectada al enlace WAN.

Se establece una cola de tipo **htb** (es una jerarquía de clases que puede conformar el tráfico a las tasas especificadas) en la interfaz interna, y una cola de tipo **sfq** (Stochastic Fairness Queueing: reordena el tráfico encolado para que cada sesión pueda enviar un paquete por turnos.) para la interfaz conectada a internet. Por cuestiones de performance [27], no se desea que el tráfico dentro de la red inalámbrica viaje a más de 20mbit.

```
#!/bin/bash
INT_IF=<interfaz con la LAN>
EXT_IF=<interfaz con Internet>
local TCQ="tc qdisc add dev ${INT_IF} "
local TCC="tc class add dev ${INT_IF} "
```

```
local TCF="tc filter add dev ${INT_IF} "
local AB="20mbit"
tc qdisc add root dev ${EXT_IF} sfq \
perturb 10
${TCQ} root handle 1: htb default 30 r2q 20
${TCC} parent 1: classid 1:1 htb rate ${AB}\
ceil ${AB} burst 2k
${TCC} parent 1:1 classid 1:10 htb rate 7mbit\
ceil 10mbit burst 2k
${TCC} parent 1:1 classid 1:20 htb rate 1mbit\
ceil 2mbit burst 2k
${TCC} parent 1:1 classid 1:30 htb rate \
512kbit ceil 1mbit burst 2k
${TCQ} parent 1:10 handle 10: sfq perturb 10
${TCQ} parent 1:20 handle 20: sfq perturb 10
${TCQ} parent 1:30 handle 30: sfq perturb 10
${TCF} protocol ip parent 1:0 prio 1 handle 1\
fw flowid 1:10
${TCF} protocol ip parent 1:0 prio 1 handle 2\
fw flowid 1:20
```

Para etiquetar el tráfico:

```
iptables -A OUTPUT -t mangle -o ${INT_IF} \
-p tcp --match multiport --sports \
3128, 443, 53,123 -j MARK --set-mark 0x1
iptables -A OUTPUT -t mangle -o ${INT_IF} -p \
tcp --match multiport --sports \
110,143,993,995 -j MARK --set-mark 0x2
```

Según estos comandos, el tráfico que proviene del proxy, o pertenece a los protocolos https, dns o ntp, lo etiquetamos con un “1” y lo enviamos a la qdisc 1:10, que tiene asegurados 7mbits y un límite superior de 10mbits. La mayoría de este tráfico va a provenir del proxy.

Lo ideal sería estimar con precisión cuánto del tráfico web que se consume sale del proxy (hit) y cuánto no estaba en el proxy y hubo que pedirlo (miss) de acuerdo a los monitoreos realizados. Si la cola se configura con “exactamente” el ancho de banda del enlace wan, se asegura que no haya problemas de congestión en la interfaz de salida a internet, pero se pierde el plus de ancho de banda que nos provee el proxy cuando hizo hit. Si se sobreestima demasiado este número, la configuración de QoS no sirve, porque los paquetes se encuentran en una situación de cuello de botella en la que no se van a encolar en nuestro servidor de QoS, pero si en el enlace a Internet que está detrás de la interfaz de salida del server.

Con la segunda regla de iptables, si el tráfico proviene de servicios pop3, imap, pop3s o imaps, lo marcamos con un “2” y lo encolamos en la cola 1:20, que tiene 1mbit reservado.

Todo el tráfico restante, va a parar a la cola 1:30, que tiene un 512K reservados, y puede trepar hasta 1 mega, si el enlace está ocioso.

Con esto, se busca que un cliente conectado a la red wifi no pueda abusar del servicio mediante aplicaciones p2p, pues -a los sumo- podrá bajar información a 1mbit y esto si nadie en toda la red, está usando ningún protocolo similar.

4 Reportes

El demonio ***mrtg*** recolecta estadísticas periódicas de uso de los enlaces vía SNMP (Fig. 3), y construye gráficas que indican tráfico total entrante y saliente, sin discriminar por tipo de protocolo. Las gráficas se generan dinámicamente a través de un cgi en el servidor web. En estos gráficos se pueden ver qué zonas están utilizando el servicio, cuánto ancho de banda se está consumiendo, en qué horarios se registran los picos de uso y también cómo el tráfico de cada terminal se va agregando en el equipo central del mesh.

El conjunto de aplicaciones ***flowscan*** (netflow) recolectan información en el servidor central sobre el número de flujos que enruta, registrando su tipo y tamaño (Fig. 4 y Fig. 5).

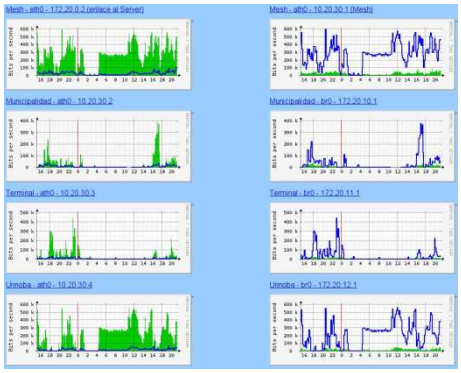

Fig. 3

Si la red está congestionada, o hay problemas en los enlaces, la latencia sube. Para monitorear esta variable, utilizamos ***smokeping***, que envía paquetes ICMP periódicamente, registrando los tiempos de respuesta. La herramienta ***sarg*** realiza reportes diarios, semanales y mensuales del uso del proxy, registrando usuarios (IP), destinos, horarios, tamaño de flujo. La herramienta ***icinga*** (fork de nagios), muestra el estado actual de los enlaces y del servidor, y puede enviar mails y sms si detecta alguno de los nodos caídos. La herramienta ***munin*** entrega variada información sobre el estado del servidor, entre otras: uso de memoria y carga de CPU del server, número de procesos y cantidad de accesos al servidor web, I/O sobre disco, latencia, %uso, temperatura, cantidad de conexiones que pasan por el firewall, errores sobre interfaces, tasa de tráfico sobre las interfaces de red, conexiones administradas, estado de la caché del proxy, cantidad de clientes y de tráfico que gestionó.

5 Conclusiones

Se propuso realizar un análisis del problema del tráfico creciente en redes de datos, el aumento en los requerimientos sobre el ancho de banda de los enlaces y el problema de congestión y pérdida de calidad de servicio; y se presentan herramientas

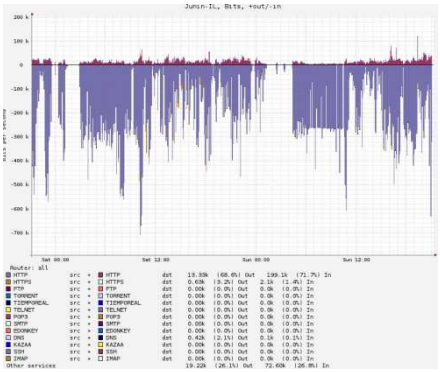

Fig. 4

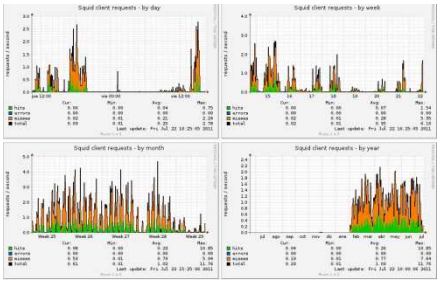

Fig. 5

que permiten gestionar estos problemas, mediante la implementación de *traffic shaping y traffic policing* en GNU/Linux en un escenario real en donde se provee de acceso libre a internet en espacios públicos en la ciudad de Junín abarcando abarcando la plaza principal de Junín -"25 de mayo"-, las plazas aledañas a la terminal de ómnibus y la plaza Ferrocarriles Argentinos utilizando herramientas de software libre disponibles para implementar la Gestión de QoS.

En el desarrollo de esta implementación se pudo:

Analizar el problema del tráfico creciente en redes de datos y la necesidad de identificar y privilegiar un determinado tipo de tráfico.

Estudiar las distintas soluciones que existen para lograr Control de Tráfico y Calidad de Servicio.

Analizar en detalle la implementación de colas para lograr QoS y comparar las distintas clases y variantes que se han desarrollado.

Analizar las herramientas que provee GNU/Linux para administrar Colas de tráfico

Estudiar el desempeño de las colas HTB en escenarios de laboratorio y reales.

Las herramientas provistas por GNU/Linux son maduras, documentadas y a lo largo de estos últimos años, ampliamente probadas.

La implementación inicial de soluciones de este tipo es relativamente sencilla. El trabajo real es ajustarlas para que cumplan lo mejor posible los objetivos para los cuales se desarrolló. Si se cuenta con un ancho de banda saliente de 5 megabytes, y luego de analizar la tasa de aciertos y fallos del proxy se ve que es cercana al 50%, deben ajustarse los valores estimados a priori sobre cuál debería ser el ancho de banda total de tráfico que puede consumir la red inalámbrica. Suponiendo que el 50% de las solicitudes de la red las resuelve el proxy, se configuran 10 Mbps para las colas de QoS. Si la tasa de aciertos cae, significa que de los 10 Mbps que llegan al proxy, más de 5 Mbps de datos debe ir a buscarlos a internet. Si el enlace es de 5 Mbps, se forma un cuello de botella y toda la infraestructura de colas (que está antes del cuello de botella mencionado) se vuelve inoperante. Entonces, o se amplía el ancho de banda del enlace WAN, o se reduce el límite superior de tráfico que se permite enviar a la red de clientes. Por eso en estas implementaciones, es necesario monitorear e ir adecuando las configuraciones periódicamente.

La configuración del proxy administra eficientemente el ancho de banda por cliente, y las colas HTB hacen lo propio con el resto del tráfico. Para un volumen de tráfico mediano, la solución planteada es eficaz y económica y se encuentra funcionando desde diciembre de 2010.

6 Bibliografía

1. Stallings, William. "Redes e Internet de Alta Velocidad. Rendimiento y Calidad de Servicio", pág 16. ISBN: 978-84-205-3921-8, Prentice Hall.
2. Internetwork Operating System, Cisco Systems.
3. "Architecture for Differenciated Services http://tools.ietf.org/rfc/rfc2475.txt
4. Busybox (http://www.busybox.net/about.html),
5. http://www.debian.org
6. http://www.netfilter.org/
7. http://www.linux-foundation.org/en/Net:Iproute2
8. http://www.squid-cache.org/
9. http://oss.oetiker.ch/mrtg/
10. http://oss.oetiker.ch/rrdtool/
11. http://www.wireshark.org/
12. http://iptraf.seul.org/
13. http://oss.oetiker.ch/smokeping/
14. http://www.caida.org/tools/utilities/flowscan/
15. http://www.icinga.org
16. http://riutec.riu.edu.ar/lib/exe/fetch.php?media=wlc-slides.pdf
17. http://www.aui.edu.ar/
18. http://www.cisco.com/en/US/prod/collateral/wireless/ps5679/ps8368/data_sheet_c78-532987.html
19. http://www.cisco.com/en/US/products/ps7221/index.html, con software de control WCS:
20. http://www.cisco.com/en/US/products/ps6305/index.html
21. http://www.wavionnetworks.com/
22. http://www.planet.com.tw/en/product/product_ov.php?id=5774
23. http://wiki.squid-cache.org/Features/DelayPools
24. Véase el manual de tc para las qdisc en http://linux.die.net/man/8/tc
25. http://es.wikipedia.org/wiki/IEEE_802.11#802.11a