Monitorización MRTG en servidores Debian Woody

Ignacio Barrancos Martínez
ignacio@adesx.com

Contenido

1. Introducción

MRTG es el abrevio de *Multi Router Traffic Grapher*, y es un simple script OpenSource escrito en perl, capaz de ejecutar un script o una consulta SNMP, para luego dibujar una gráfica con los resultados históricos de las consultas. Para dibujar las gráficas se basa en GD.

GD es una librería en ANSI C desarrollada por Thomas Boutell, que permite la creación dinámica de imágenes en varios formatos como PNG y JPEG, aunque entre ellos no se incluye la creación de GIFs. Dispone de distintos ports a diferentes lenguajes de programación como perl o php. En concreto, MRTG requiere del wrapper para perl llamado GD.pm.

SNMP es el abrevio de *Simple Network Management Protocol*, y es un protocolo para la manipulación, configuración y consulta de dispositivos de red. En la actualidad el 99% de los dispositivos que podemos conectar a nuestra red (impresoras, switches, routers), así como servidores (*Novell*, *Windows*, *Linux*, *HP/UX*, etc...) son capaces por lo menos, de informar de su estado a través de una consultas SNMP.

Cualquiera de estos elementos de red, capaces de responder vía SNMP, disponen de un *agente SNMP* que implementa toda una jerarquía de variables conocida como **MIB** (*Management Information Base*): Así, una impresora tendrá una jerarquía (*MIB*) diferente a la que pueda tener un switch, o un servidor. Incluso dos modelos de impresora diferentes de una misma marca podrán tener MIB diferentes.

Gracias a que la especificación de los MIB se recoge en varios estandares y RFCs, los distintos fabricantes están obligados a implementar determinadas ramas de la jerarquía, para mantener cierta compatibilidad básica con las herramientas de consulta, que de otro modo resultaría imposible: *antes de consultar nada, se debería conocer al 100% todo el MIB de todos los dispositivos*. Así, los fabricantes implementan una jerarquía básica que luego complementan con sus mejoras, y que es distinta del fabricante y del tipo de elemento del que se trate.

Estos MIB son simples ficheros en texto plano, cuya sintaxis viene expresada en ASN.1 (*Abstract Syntax Notation One*). ASN.1 facilita la comunicación entre profesionales y miembros de un comité, ofreciéndoles un idioma común para describir un estándar, y fué diseñado para eso mismo: proveer un lenguaje para la especificación de estándares.

En la actualidad existen buscadores específicos de MIBs que pueden resultarnos de mucha utilidad como mibdepot o somix, si se quieren monitorizar otros elementos/servidores de red, que en este documento no describiremos.

En este documento se describe cómo instalar y configurar el agente SNMP y MRTG sobre Debian Woody para monitorizar:

1. La carga de CPU de nuestro linux.
2. Tráfico de red.
3. Temperatura del procesador.
4. Carga de la memoria.
5. Procesos.
6. Ocupación de los discos/ particiones.
7. Número de conexiones de mlDonkey :-).

2. Instalación de software en Debian Woody desde apt

Lo primero que se debería instalar es **el agente y el cliente SNMP**. Para ello, y gracias a la ayuda de Francisco Jesús Rubio Reales, más conocido por *rubio*, procederemos de la siguiente manera:

```
su -apt-get install snmp snmpd
```

Luego, se editará el fichero `/etc/init.d/snmpd.conf`, y en la líneas

```
#sec.name source community
com2sec paranoid default public
#com2sec readonly default public
#com2sec readwrite default private
```

se comentará el nivel `paranoid` y descomentará `readonly`, quedando algo tal que así:

```
#sec.name source community
#com2sec paranoid default public
com2sec readonly default public
#com2sec readwrite default private
```

Se reinicia el servicio ...

```
# /etc/init.d/snmpd start
Starting network management services: snmpd snmptrapd.
```

Observaremos que se han arrancado dos servicios: `snmpd` que es el agente SNMP, que permite que nuestro Linux responda a las peticiones que se le realicen por el puerto 161 de UDP, y otro servicio llamado `snmpdtrad`, que permite tener un demonio que es avisado cuando ocurre determinado evento en nuestra subred que no configuraremos aquí.

En principio, ya podremos probar a lanzar una consulta desde consola a ver qué pasa:

```
snmpwalk -c public -v 1 127.0.0.1 system
```

la pantalla se llenará de un montón de líneas, que en realidad se corresponden con cada una de las variables MIB que la implementación del agente para Debian es capaz de informar. Antes de terminar (mensaje *End of MIB*) dará un timeout, y veremos como el agente ha dejado de ejecutarse. Es el problema que yo le veo, por eso, prefiero compilarlo e instarlo a mano. Para desinstarlo ejecutaremos,

```
apt-get remove snmp snmpd --purge
```

3. Compilación e Instalación del agente snmp en Debian Woody

El agente Lo descargamos de http://heanet.dl.sourceforge.net/sourceforge/net-snmp/net-snmp-5.1.1.tar.gz , que es la última versión estable que he encontrado. La instalación la realizaremos con stow, que es un maravilloso paquete para gestionar el software que instalamos compilándolo. Para ello ...

```
wget -b \
     http://heanet.dl.sourceforge.net/sourceforge/net-snmp/net-snmp-5.1.1.tar.gz
tar -xzvf net-snmp-5.1.1.tar.gzcd net-snmp-5.1.1
./configure --prefix=/usr/local/stow/net-snmp-5.1.1
make
make install
cd /usr/local/stow/
stow net-snmp-5.1.1
```

Con esto ya lo tendremos instalado en nuestro `/usr/local`. Ahora deberíamos instalar los scripts de arranque y configuración de snmpd que tengo en mi repositorio subversion, dado que al compilarlo a mano no se crearán porque ello dependerá de nuestra distribución.

```
su -
svn checkout http://ignacio-barrancos.dnsalias.net/repositorio/debian/net-snmp
cd net-snmp
make
```

Ello nos habrá copiado el demonio snmpd en `/etc/init.d` y el archivo de configuración en `/etc/snmp/`. Este archivo lo generé con el comando `snmpconf -g basic_setup` que nos irá haciendo preguntas; se generará el archivo de configuración a partir de las respuestas que le demos en el directorio actual. Para configurar que el demonio arranque con el sistema, teclearemos como root ...

```
update-rc.d snmpd defaults 99
```

y ya podemos arrancarlo:

```
/etc/init.d/snmpd start
```

Para probar que esto funciona, ejecutaremos:

```
snmpwalk -c public -v 1 127.0.0.1 system
```

la pantalla se llenará de un montón de líneas, que en realidad se corresponden con las variables MIB que el agente es capaz de informar. Ahora podremos observar como no se cuelga el agente a diferencia de lo que sucedía con el demonio instalado con `apt`. El comando que se ha usado es `snmpwalk` (*hablar por snmp*), y los parámetros que se le han pasado son:

- `-c public` : Indica el nombre de la comunidad para la consulta. Por defecto, es la comunidad con permisos de sólo lectura.
- `-v 1` : Indica que se usará la versión 1 del protocolo (actualmente existen tres: 1, 2c y la 3).
- `127.0.0.1` : Es la Ip del elemento de red que tiene el agente SNMP ejecutándose y que quiero consultar.

- `.1.3.6.1` : Es la variable MIB que quiero conocer. En realidad esta variable en cristiano representa la consulta: *(1)iso.(3)org.(6)dot.(1)internet*

Si se quiere profundizar más sobre esto, sería aconsejable instalarnos un cliente SNMP gráfico, que permita navegar visualmente por los MIB de los distintos fabricantes. Recomiendo uno para Windows, *Nu-Design Visual MIBrowser Pro 3.1*.

4. Instalación y configuración de mrtg

Ahora se instalará `mrtg` mediante `apt`.

```
apt-get install mrtg
```

Una vez está instalado, podremos utilizar los comandos `indexmaker` y `cfgmaker` para crear un archivo básico de configuración para mrtg, pero están limitados a lo típico. Aquí nos lo vamos a currar entero, y pasaremos por completo de estos comandos.

Mrtg tiene un problema bastante grande, y es que se corrompen los ficheros de logs que él gestiona para pintar las gráficas, con suma facilidad, siendo los motivos más habituales el que el demonio snmpd tarde en responder, y de un timeout, o que se ejecute dos veces mrtg sobre el mismo archivo de configuración. Una vez se corrompen los ficheros, es casi imposible recuperarlos.

Por este motivo, he implementado un conjunto de scripts, que impedirán que se corrompan estos ficheros y darán robustez, al tiempo que presentarán los resultados con el aspecto de mi web con el uso de plantillas xsl. Lo he subido a mi repositorio subversion.

```
su -
svn checkout http://ignacio-barrancos.dnsalias.net/repositorio/mini-projects/mrtg
cd mrtg
make
```

Esto instalará el proyecto subversion en el directorio `/var/mrtg`. Si se quiere cambiar, habrá que editar los ficheros `etc/config.cfg` y `etc/mrtg.linux.cfg`, y sustituir todas las apariciones de `/var/mrtg` por el directorio donde queramos instalarlo. En principio, los scripts están pensados para que al instalarlos monitoricen un sólo equipo de nuestra red (por eso aparece 127.0.0.1 en el fichero `mrtg.linux.cfg`), pero más adelante se dirá cómo ampliarlo a más de un equipo de nuestra red.

Para que funcione la construcción de los informes en html, necesita que esté instalado `Xalan` y además tiene la dependencia del proyecto `xml` de mi repositorio . De él necesita las XSLs para la web.

Una vez lo hemos instalado, lo único que faltará será programarlo en el cron. Para ello editaremos como root, nuestro cron (`crontab -e`) y añadiremos las siguientes líneas:

```
#-Graficas MRTG
0-55/5 * * * * /var/mrtg/bin/mrtgGraph.sh /var/mrtg/etc/mrtg.linux.cfg
```

Con lo que le estamos diciendo que genere las gráficas cada 5 minutos durante todo el día, todos los días de la semana.

4.1. Configuración de los scripts

La configuración de todos los scripts que he desarrollado está en el subdirectorio `etc/` y se concentra en los siguientes ficheros:

- **`config.cfg`** : Contiene la ruta donde se almacenarán los logs y gráficas de mrtg, dónde se publicarán en html los resultados, y las rutas absolutas de todos los comandos que los scripts utilizan, para

evitar problemas de configuración con el entorno.

- **`mrtg.linux.cfg`** : Es el archivo de configuración para mrtg, que este usará para obtener los datos con los que pintará las gráficas.

Destacar que en él he configurado para monitorizar:

1. Tráfico de red de eth0, en cuanto a bytes transmitidos/recibidos por segundo. Esto me permite ver cuándo telefónica me trinca la línea ADSL, que es de vez en cuando.
2. Carga de uso de la CPU, similar al segundo parámetro que ofrece el comando `top`, dentro de los valores de carga de CPU.
3. Uso de memoria RAM y Swap, medido en tanto por ciento de ocupación. Es bueno conocer la swap porque cuando el sistema empieza a coger mucha swap, significa que pronto caerá.
4. Reparto de la CPU: Tanto por ciento de uso de los procesos del system y de los usuarios. Señalar que el equipo lo tengo configurado para que casi todos los servicios se ejecuten como usuarios unix.
5. Ocupación de las particiones del disco duro, representado en tanto porciento.
6. Número de conexiones TCP del demonio mlDonkey. Me permite observar cómo me chupan los archivos la gente de emule y bittorrent. :).
7. Temperatura del procesador. Ya llega el Verano: a ver cómo se porta con el micro y la refrigeración esta rara que tiene.

4.2. Significado de los scripts

Los scripts están en el subdirectorio `bin/` y se han modularizados para realizar las siguientes tareas:

snmp-common.sh : Contiene de manera centralizada la llamada a snmpget, que es el comando que se usará para realizar la consulta snmp. Se ha sacado aparte como una función, porque varios scripts necesitarán invocarla.

snmpGet.sh : Es el script encargado de realizar consultas snmp genéricas. Este script espera como primer parámetro la ip del agente a consultar, y luego los mibs de la consulta. Lo podemos probar con la interfaz de loopback:

```
./snmpGet.sh 127.0.0.1 1.3.6.1.2.1.2.2.1.10.1 1.3.6.1.2.1.2.2.1.16.1
```

Como se puede ver, la salida del script siempre serán cuatro líneas: La primera con el valor de la consulta para el primer mib, la segunda con el valor del segundo mib, la tercera con el uptime del equipo, y la última con el nombre del equipo. Si tu PC no se llama `helicon` , y la consulta te devolvió en la última línea que el equipo se llamaba `helicon` , edita el fichero `/etc/snmp/snmpd.conf` y reemplaza `helicon` por el nombre de tu equipo, así como `ignacio@adesx.com` por tu dirección email. Luego reinicia el agente snmp, y vuelve a comprobar la salida de `snmpGet.sh`.

El formato de salida es así, porque así lo necesita mrtg para que los ficheros de logs no se corrompan.

snmpGetMemory.sh : Este script permite obtener el tanto porciento de memoria ram y swap usada por un servidor Linux, con el agente `net-snmp-5.1.1` . El único parámetro que espera, es la dirección Ip del servidor a consultar.

snmpGetDonkey.sh : Este script devolverá el número de conexiones TCP que tiene abiertas el proceso *mlnet* (el de `mldonkey`). Ello se hace sin usar snmp, sino ejecutando el comando `netstat` con `wc` , por lo que no se podrá usar para monitorizar equipos remotos de nuestra red.

snmpGetTemperature.sh : Devuelve los grados centígrados a los que está nuestro procesador. Para ello se consulta el contenido del archivo `/proc/acpi/thermal_zone/THRM/temperature`, por lo que debemos tener compilado el kernel con soporte acpi, y nuestra placa lo debe soportar.

4.3. Ampliando la monitorización

Si se quieren monitorizar nuevos parámetros que no se han recogido aquí, deberemos tener en cuenta principalmente, si estos nuevos parámetros están disponibles vía SNMP o no. Si sabemos que están disponibles, tan sólo bastará editar el archivo `etc/mrtg.linux.cfg` , y añadir la sección de la gráfica que queremos mostrar. Si por el contrario, los parámetros no están disponibles por SNMP deberemos crearnos scripts que extraigan estos parámetros y los muestren tal y como lo hacen los scripts `bin/mrtgGetDonkey.sh` y `bin/mrtgGetTemperature.sh`: *La primera línea el valor 1, la segunda el valor 2, la tercera el uptime, y la cuarta el nombre del equipo.*

Si en vez de ampliar los parámetros, lo que se quiere es monitorizar de forma centralizada más equipos desde un único equipo, bastará con crear nuevos `etc/mrtg.linux.cfg` (con otro nombre), donde no se repitan los nombres de los *targets* (red, disk, mem, etc...), porque al convertirlos en XML se usa este nombre para generar los ficheros Html resultantes (valdrán, por ejemplo nombres como, red_goku, red2, mem7, etc...). También se deberá crear un nuevo `index.html` en la ruta de publicación de las gráficas, que enlace a cada uno de los informes diarios. Se añade una línea al cron para llamar con el nuevo cfg y a correr.

El monitorizar equipos de forma centralizada, nos limita a que sólo podremos usar SNMP para consultar el valor de los parámetros, siendo inviable la posibilidad de ejecutar scripts del estilo de "`bin/mrtgGetDonkey.sh`", aunque siempre se podría modificar el demonio de *ucdavis*, o currarse mucho el "`/etc/snmp/snmpd.conf`", para monitorizar esto.

Si se desea cambiar el formato de presentación de los resultados, bastará con modificar el fichero `xsl/global.xs` y personalizarlo a nuestro gusto y disgusto.

Que ustedes lo disfruten.