Apéndice D

Tratamiento de la información

Índice del capítulo

D.1. Tratamiento de la información de *NetFlow*

D.1.1. Presentación mediante consola

Una vez que tenemos la información recogida por el recolector *flow-capture* está almacenada en nuestro sistema de monitorización, podemos pasar a presentarla. En este primer apartado trataremos la simple presentación mediante consola.

Para ello vamos a utilizar algunas de las utilidades incluidas en el paquete ***flow-tools***, ya instalado en el apéndice A.3.1, "*Instalación de flow-tools*", concretamente las utilidades *flow-cat* y *flow-print*.

flow-cat es la herramienta que permite leer de manera recursiva y ordenada los archivos emplazados dentro de la estructura de directorios que ha creado el recolector. Leerá los datos comprimidos del disco almacenados por *flow-capture* y su salida es de un formato intermedio que ha de ser redirigido a las otras utilidades del paquete ***flow-tools*** para que operen con los datos. La segunda de las utilidades, *flow-print*, es la que tomará esa información en formato intermedio y la mostrará en formato ASCII por la consola.

Veamos algunas de los parámetros más interesantes que presentan ambas utilidades. Los de *flow-cat* los mostramos en el Cuadro D.1 y los de *flow-print*, que dispone de muchas menos opciones, en el Cuadro D.2. Para más información sobre estas aplicaciones pueden consultarse sus páginas del manual [16, 17].

Parámetro	**Descripción**
-t `start_time`	Hora de comienzo del intervalo que se quiere estudiar.
-T `end_time`	Hora de final del intervalo.
-a	No ignorará los archivos temporales emplazados en la estructura de directorios.
`file|directory`	Archivo o ruta del directorio a analizar. En el caso del directorio es de comportamiento recursivo.

Cuadro D.1: Parámetros de *flow-cat*

Parámetro	**Descripción**
-f `format`	Formato de la salida. Algunos de ellos son: 0 Una línea: muestra interfaces, IPs, protocolos, cantidad de paquetes y octetos, y los puertos en hexadecimal. Muy abreviada. 3 Una línea: como la anterior, pero suprime los interfaces para muestrar los puertos en decimal. Más legible. 6 Una línea: muestra solo IPs y cantidad de paquetes y octetos. Útil para controlar consumos. 24 Salida completa: IPs, interfaces, puertos, protocolos, cantidad de paquetes y octetos, exportador. Muy detallada. Por defecto *flow-print* se adaptará a la información que deba visualizar.
-n	Usar nombres simbólicos donde sea adecuado si el formato de salida es `3`. Configurable en `/etc/flow-tools/sym/`.

Cuadro D.2: Parámetros de *flow-print*

En este momento es necesario recordar que la información que se muestra es información generada por la sonda, y que las capacidades de las herramientas del paquete ***flow-tools*** son directamente dependientes de la información de la sonda. Por lo tanto, la información que la sonda no inserte o inserte indebidamente en el flujo *NetFlow* será interpretada de forma directa por el recolector. Veamos pues cual es la información y el formato que es mostrado por *flow-print*:

```
#> flow-cat /var/flow/eth0/ | flow-print

srcIP               dstIP               router_sc        prot    srcPort
     dstPort         octets      packets
```

```
193.189.124.15/0    192.168.100.24/0    0.0.0.0          6      80
    32780           7756         10
192.168.100.24/0    193.189.124.15/0    0.0.0.0          6      32780
    80              1526         10
192.168.100.150/0   192.168.100.255/0   0.0.0.0          17     138
    138             237          1
192.168.100.150/0   192.168.100.255/0   0.0.0.0          17     138
    138             244          1
150.214.186.69/0    192.168.100.24/0    0.0.0.0          17     53
    32769           393          3
192.168.100.24/0    150.214.186.69/0    0.0.0.0          17     32769
    53              176          3
```

En cambio, podemos pedir a *flow-print* que nos muestre la misma información más detallada:

```
#> flow-cat /var/flow/eth0/ | flow-print -f 24

#Sif SrcIPaddress     DIf  DstIPaddress     Pr SrcP  DstP  Pkts       Octets
          StartDate  StartTime     EndDate    EndTime       ExporterAddr
    RouterSrc          Active B/Pk  Ts Fl  SrcAS DstAS

0000 193.189.124.15   0000 192.168.100.24   06 80    32780 10         7756
           2005-11-25 17:45:57.500  2005-11-25 17:46:26.040  192.168.100.24
    0.0.0.0             28.540 775   00 1b  0     0
0000 192.168.100.24   0000 193.189.124.15   06 32780 80    10         1526
           2005-11-25 17:45:57.455  2005-11-25 17:46:25.997  192.168.100.24
    0.0.0.0             28.542 152   00 1b  0     0
0000 192.168.100.150  0000 192.168.100.255  11 138   138   1          237
            2005-11-25 17:51:44.989  2005-11-25 17:51:44.989  192.168.100.24
    0.0.0.0              0.000 237   00 00  0     0
0000 192.168.100.150  0000 192.168.100.255  11 138   138   1          244
            2005-11-25 17:57:54.504  2005-11-25 17:57:54.504  192.168.100.24
    0.0.0.0              0.000 244   00 00  0     0
0000 150.214.186.69   0000 192.168.100.24   11 53    32769 3          393
            2005-11-25 17:57:40.875  2005-11-25 17:58:14.303  192.168.100.24
    0.0.0.0             33.428 131   00 00  0     0
0000 192.168.100.24   0000 150.214.186.69   11 32769 53    3          176
            2005-11-25 17:57:40.868  2005-11-25 17:58:14.296  192.168.100.24
    0.0.0.0             33.428 58    00 00  0     0
```

Por último, podemos pedir a *flow-cat* que nos muestre sólo un intervalo de tiempo concreto a modo de filtrado temporal:

```
#> flow-cat -t "11/27/05 11:00" -T "11/27/05 11:15" /var/flow/eth0/ | flow-print
    -f 24

#Sif SrcIPaddress     DIf  DstIPaddress     Pr SrcP  DstP  Pkts       Octets
          StartDate  StartTime     EndDate    EndTime       ExporterAddr
    RouterSrc          Active B/Pk  Ts Fl  SrcAS DstAS

0000 192.168.100.150  0000 192.168.100.255  11 138   138   1          237
            2005-11-27 11:03:45.345  2005-11-27 11:03:45.345  192.168.100.24
    0.0.0.0              0.000 237   00 00  0     0
0000 192.168.100.22   0000 192.168.100.24   06 42350 514   1          112
            2005-11-27 11:06:58.913  2005-11-27 11:06:58.913  192.168.100.24
    0.0.0.0              0.000 112   00 18  0     0
0000 192.168.100.24   0000 192.168.100.22   06 514   42350 1          52
             2005-11-27 11:06:58.914  2005-11-27 11:06:58.914  192.168.100.24
    0.0.0.0              0.000 52    00 10  0     0
0000 192.168.100.150  0000 192.168.100.255  11 138   138   1          244
            2005-11-27 11:12:53.562  2005-11-27 11:12:53.562  192.168.100.24
    0.0.0.0              0.000 244   00 00  0     0
```

D.1.1.1. Filtrando la información de *NetFlow*

Existen otras herramientas incluidas en el paquete ***flow-tools*** que permiten hacer estudios más afinados que los anteriormente presentados. *flow-nfilter* es una utilidad que permite añadir filtros para seleccionar con más detalle la

información que analizamos. La definición de los filtros se basa en la creación de unos ficheros de configuración que posteriormente leerá *flow-nfilter* . En el Cuadro D.3 se muestran algunos de los parámetros principales de *flow-nfilter*. Para más detalle puede consultarse el manual de la utilidad [18].

Parámetro	**Descripción**
-f `filter_fname`	Fichero que contiene las definiciones de los filtros. Por defecto usará `/etc/flow-tools/cfg/filter`.
-F `filter_definition`	Filtro a utilizar. Por defecto utilizará el filtro `default`.

Cuadro D.3: Parámetros de *flow-nfilter*

La forma de usarlo es sencilla: se filtrará la salida de *flow-cat* hacia la siguiente utilidad que estemos utilizando para analizar la información. Los parámetros para la configuración de *flow-nfilter* y la formación de sus filtros son muchos y referimos al lector a su manual para conocerlos con detalle.

En el siguiente ejemplo hemos definido un archivo llamado `filtro.cfg` con el siguiente contenido:

`/etc/flow-tools/cfg/filtro.cfg`

```
filter-primitive port80
         type ip-port
         permit 80

filter-primitive port3000
         type ip-port
         permit 3000

filter-definition FILTRADO
         match ip-source-port port80
         or
         match ip-destination-port port3000
```

Tras ello, basta solo utilizar el comando adecuado. En este caso, a modo de ejemplo, utilizamos también las capacidades de filtrado temporal de *flow-cat* para mostrar la plena compatibilidad entre aplicaciones:

```
#> flow-cat -t "11/08/05 19:00" -T "11/08/05 19:15" /var/flow/eth0/ | flow-
    nfilter -f /etc/flow-tools/cfg/filtro.cfg -F FILTRADO | flow-print

192.168.100.24/0    192.168.100.22/0    0.0.0.0          6      32886
    3000            1071        11
192.168.100.24/0    192.168.100.22/0    0.0.0.0          6      32885
    3000            860         6
192.168.100.24/0    192.168.100.22/0    0.0.0.0          6      32887
    3000            1049        10
192.168.100.24/0    192.168.100.22/0    0.0.0.0          6      32888
    3000            1124        11
192.168.100.24/0    192.168.100.22/0    0.0.0.0          6      32889
    3000            822         7
204.152.191.5/0     192.168.100.24/0    0.0.0.0          6      80
    32890           3184381     2129
192.168.100.24/0    192.168.100.22/0    0.0.0.0          6      32906
    3000            801         6
192.168.100.24/0    192.168.100.22/0    0.0.0.0          6      32903
    3000            762         5
192.168.100.24/0    192.168.100.22/0    0.0.0.0          6      32904
    3000            761         5
```

```
192.168.100.24/0    192.168.100.22/0    0.0.0.0          6       32902
    3000            761          5
192.168.100.24/0    192.168.100.22/0    0.0.0.0          6       32895
    3000            752          5
192.168.100.24/0    192.168.100.22/0    0.0.0.0          6       32898
    3000            761          5
192.168.100.24/0    192.168.100.22/0    0.0.0.0          6       32899
    3000            761          5
192.168.100.24/0    192.168.100.22/0    0.0.0.0          6       32894
    3000            802          6
192.168.100.24/0    192.168.100.22/0    0.0.0.0          6       32900
    3000            761
```

D.1.1.2. Presentanto informes en la consola

Por último, la utilidad *flow-stat* contenida en el paquete ***flow-tools*** permite realizar pequeños informes muy legibles de forma cómoda. Algunos de sus parámetros más utilizados son mostrados en el Cuadro D.4. Más detalle sobre estos y otros parámetros puede consultarse en su manual [19].

Parámetro	**Descripción**
-f `format`	Formato del informe: 7 Puerto UDP/TCP. 5 Puerto UDP/TCP de destino. 6 Puerto UDP/TCP de origen. 10 IP origen/destino. 8 IP destino. 9 IP origen.

Cuadro D.4: Parámetros de *flow-stat*

Su uso es sencillo: será el destino de la salida de *flow-cat*, por lo que también podrían utilizarse filtros intermedios definidos con *flow-nfilter*:

```
#> flow-cat  /var/flow/eth0/ | flow-stat -f 10

#  --- ---- ---- Report Information --- --- ---
#
# Fields:     Total
# Symbols:    Disabled
# Sorting:    None
# Name:       Source/Destination IP
#
# Args:       flow-stat -f 10
#
#
# src IPaddr      dst IPaddr        flows                 octets
    packets
#
192.168.100.21    192.168.100.255   13                    6478                  26
192.168.100.22    192.168.100.24    678                   9953371
    26508
192.168.100.24    192.168.100.22    739                   33897308
    38038
192.168.100.23    192.168.100.24    6                     11640                 94
192.168.100.24    192.168.100.23    6                     12224                 168
150.214.186.69    192.168.100.24    15                    7326                  39
192.168.100.24    150.214.186.69    15                    2435                  39
192.168.100.24    204.152.191.5     1                     75241
    1331
```

```
204.152.191.5     192.168.100.24    1                 3184381
    2129
192.168.100.24    216.239.59.99     6                 6945              38
216.239.59.103    192.168.100.24    2                 890               6
192.168.100.24    66.35.250.209     31                37095             277
66.35.250.209     192.168.100.24    31                237410            285
192.168.100.24    67.19.167.98      2                 1612              14
66.35.250.203     192.168.100.24    2                 11484             14
67.19.167.98      192.168.100.24    2                 6261              12
192.168.100.24    66.35.250.203     2                 2182              15
18.7.14.127       192.168.100.24    2                 6482              12
192.168.100.24    18.7.14.127       2                 1530              14
64.20.33.131      192.168.100.24    2                 2954              13
192.168.100.24    64.20.33.131      2                 3145              16
```

D.1.2. Exportando a *ntop*

Veremos en esta punto cómo exportar la información recogida en el recolector a la utilidad *ntop*, que permitirá el estudio de la información de forma gráfica y cómoda.

La exportación se puede realizar de dos formas distintas: volcando periódicamente la información almacenada por el recolector o haciendo un volcado «en tiempo real».

D.1.2.1. Volcado con emisión periódica

Con el primero de los sistemas se realizaría una emisión periódica de la información recogida por el recolector *flow-capture* emitiéndose dicha información usando el protocolo *NetFlow*, que también es capaz de interpretar *ntop*.

Haremos las siguientes suposiciones acerca de la estructura de la red para este apartado:

- El sistema que tiene *ntop* preparada para recibir los flujos de *NetFlow* se encuentra en la dirección 192.168.100.24.
- El puerto en el que atenderá los mensajes será el puerto 2055.

Para la emisión de forma periódica de la información del recolector hemos usado la utilidad *flow-send* del paquete ***flow-tools***. La herramienta permite la utilización de los parámetros mostrados en el Cuadro D.5. Para más detalle puede consultarse su manual [20].

Como puede verse esto es, a todos los efectos, convertir al recolector *flow-capture* a su vez en la sonda de otro recolector, sonda que enviará la información almacenada por el recolector. Con el siguiente comando podríamos exportar ya la información hacia el nuevo recolector:

```
#> flow-cat /var/flow | flow-send -V 7 192.168.100.24/192.168.100.24/2055
```

Mostramos la Figura D.1 para comprender mejor su idea de funcionamiento.

Es importante resaltar que *ntop* recogerá la información como si de datos capturados por él mismo se tratase, de manera que si enviamos dos veces la misma información hacia *ntop* esta se verá registrada por duplicado. Para evitar

Parámetro	Descripción
-V `pdu_version`	Especifica la versión del protocolo a utilizar.
-m `privacy_mask`	Permite poner una máscara para ocultar los hosts de la red. Alterará los campos de IP de destino e IP de origen del flujo de *NetFlow*.
`localip/remoteip/port`	Permite especificar la IP del origen del flujo de datos *NetFlow*, la IP del recolector y el puerto donde este espera el flujo.

Cuadro D.5: Parámetros de *flow-send*

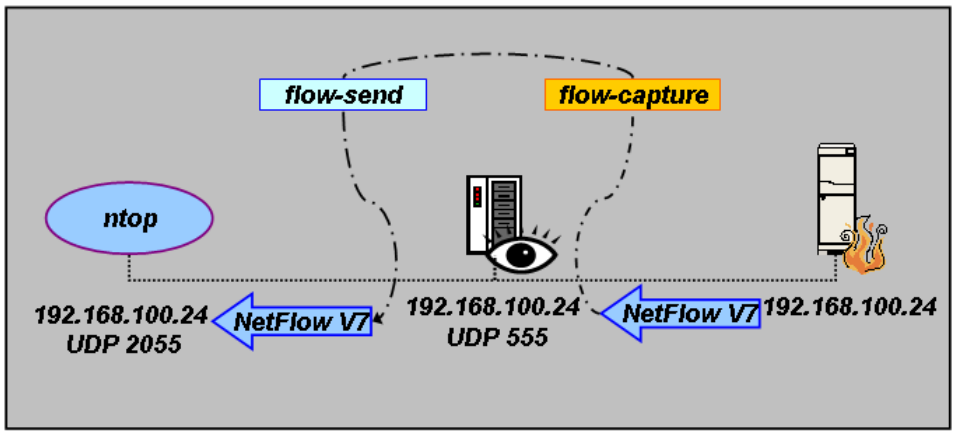

Figura D.1: Esquema de funcionamiento de *flow-send* para la exportación a *ntop*

este comportamiento podemos aprovechar el uso de la capacidad de *flow-cat* para reportar intervalos de tiempo. Presentamos a continuación un `crontab` para el demonio *cron* que emitiría la información cada hora hacia *ntop*, sin repeticiones:

`crontab`

```
    01 01 * * * flow-cat -t "00:00 Today" -T "01:00 Today" /var/flows/eth0/ | flow-
        send -V 7 192.168.100.24/192.168.100.24/2055
    01 02 * * * flow-cat -t "01:00 Today" -T "02:00 Today" /var/flows/eth0/ | flow-
        send -V 7 192.168.100.24/192.168.100.24/2055
    01 03 * * * flow-cat -t "02:00 Today" -T "03:00 Today" /var/flows/eth0/ | flow-
        send -V 7 192.168.100.24/192.168.100.24/2055
    01 04 * * * flow-cat -t "03:00 Today" -T "04:00 Today" /var/flows/eth0/ | flow-
        send -V 7 192.168.100.24/192.168.100.24/2055
 5  01 05 * * * flow-cat -t "04:00 Today" -T "05:00 Today" /var/flows/eth0/ | flow-
        send -V 7 192.168.100.24/192.168.100.24/2055
    01 06 * * * flow-cat -t "05:00 Today" -T "06:00 Today" /var/flows/eth0/ | flow-
        send -V 7 192.168.100.24/192.168.100.24/2055
    01 07 * * * flow-cat -t "06:00 Today" -T "07:00 Today" /var/flows/eth0/ | flow-
        send -V 7 192.168.100.24/192.168.100.24/2055
    01 08 * * * flow-cat -t "07:00 Today" -T "08:00 Today" /var/flows/eth0/ | flow-
        send -V 7 192.168.100.24/192.168.100.24/2055
    01 09 * * * flow-cat -t "08:00 Today" -T "09:00 Today" /var/flows/eth0/ | flow-
        send -V 7 192.168.100.24/192.168.100.24/2055
10  01 10 * * * flow-cat -t "09:00 Today" -T "10:00 Today" /var/flows/eth0/ | flow-
        send -V 7 192.168.100.24/192.168.100.24/2055
    01 11 * * * flow-cat -t "10:00 Today" -T "11:00 Today" /var/flows/eth0/ | flow-
        send -V 7 192.168.100.24/192.168.100.24/2055
    01 12 * * * flow-cat -t "11:00 Today" -T "12:00 Today" /var/flows/eth0/ | flow-
```

```
        send -V 7 192.168.100.24/192.168.100.24/2055
    01 13 * * * flow-cat -t "12:00 Today" -T "13:00 Today" /var/flows/eth0/ | flow-
        send -V 7 192.168.100.24/192.168.100.24/2055
    01 14 * * * flow-cat -t "13:00 Today" -T "14:00 Today" /var/flows/eth0/ | flow-
        send -V 7 192.168.100.24/192.168.100.24/2055
15  01 15 * * * flow-cat -t "14:00 Today" -T "15:00 Today" /var/flows/eth0/ | flow-
        send -V 7 192.168.100.24/192.168.100.24/2055
    01 16 * * * flow-cat -t "15:00 Today" -T "16:00 Today" /var/flows/eth0/ | flow-
        send -V 7 192.168.100.24/192.168.100.24/2055
    01 17 * * * flow-cat -t "16:00 Today" -T "17:00 Today" /var/flows/eth0/ | flow-
        send -V 7 192.168.100.24/192.168.100.24/2055
    01 18 * * * flow-cat -t "17:00 Today" -T "18:00 Today" /var/flows/eth0/ | flow-
        send -V 7 192.168.100.24/192.168.100.24/2055
    01 19 * * * flow-cat -t "18:00 Today" -T "19:00 Today" /var/flows/eth0/ | flow-
        send -V 7 192.168.100.24/192.168.100.24/2055
20  01 20 * * * flow-cat -t "19:00 Today" -T "20:00 Today" /var/flows/eth0/ | flow-
        send -V 7 192.168.100.24/192.168.100.24/2055
    01 21 * * * flow-cat -t "20:00 Today" -T "21:00 Today" /var/flows/eth0/ | flow-
        send -V 7 192.168.100.24/192.168.100.24/2055
    01 22 * * * flow-cat -t "21:00 Today" -T "22:00 Today" /var/flows/eth0/ | flow-
        send -V 7 192.168.100.24/192.168.100.24/2055
    01 23 * * * flow-cat -t "22:00 Today" -T "23:00 Today" /var/flows/eth0/ | flow-
        send -V 7 192.168.100.24/192.168.100.24/2055
    01 00 * * * flow-cat -t "23:00 Yesterday" -T "00:00 Today" /var/flows/eth0/ |
        flow-send -V 7 192.168.100.24/192.168.100.24/2055
```

D.1.2.2. Volcado «en tiempo real»

Este segundo método hará que la información recogida por el recolector sea retransmitida al momento hacia *ntop*, por lo que la calificamos «en tiempo real».

En este apartado haremos también las siguientes suposiciones acerca de la estructura de la red :

- El sistema que tiene *ntop* preparado para recibir los flujos de *NetFlow* se encuentra en la dirección IP 192.168.100.24.
- Dicho sistema espera recibir los mensajes en el puerto 2055.

Para esta segunda forma de emisión usaremos la herramienta *flow-fanout* incluida también en el paquete ***flow-tools***. Se muestra la Figura D.2 para ilustrar el uso que se hará de *flow-fanout*.

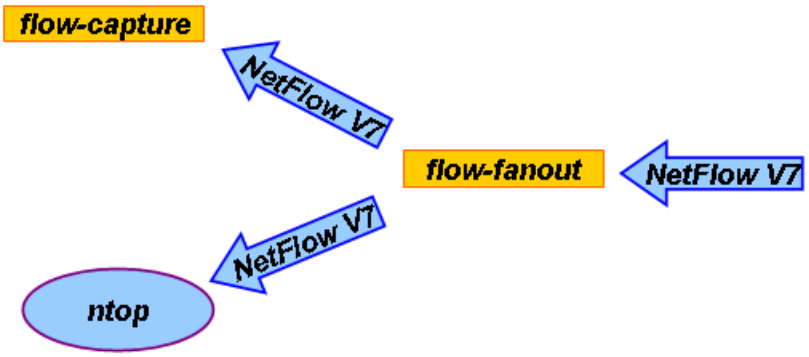

Figura D.2: Esquema de funcionamiento de *flow-fanout* para la exportación a *ntop*

El funcionamiento mostrado de esta herramienta hará que tengamos que modificar la configuración de *flow-capture* usada hasta ahora, dado que el flujo de los paquetes de *NetFlow* será el siguiente:

1. El flujo de la sonda del sistema remoto será enviado hacia *flow-fanout*.
2. *flow-fanout* retransmitirá el flujo hacia nuestro recolector, *flow-capture*.
3. *flow-fanout* retransmitirá igualmente el flujo hacia *ntop*.

Los parámetros de configuración que *flow-fanout* acepta son los indicados en el Cuadro D.6. Se pueden consultar más detalles de esta aplicación en su manual [21].

Dicho esto, y modificando la configuración de *flow-capture* para que atienda los flujos en el puerto 556 en vez de en 555, bastaría el siguiente comando para que *flow-fanout* enviase la información tanto a *flow-capture* como a *ntop* si este está escuchando en la máquina 192.168.100.24 en el puerto 2055:

```
#> flow-fanout 192.168.100.24/192.168.100.24/555
     192.168.100.24/192.168.100.24/556 192.168.100.24/192.168.100.24/2055
```

Sea cual sea el sistema escogido, una vez se dispone de la información exportada a *ntop* éste se puede manejar idénticamente a como se manejaría en el caso de captura normal, como se muestra a modo de ejemplo en la Figura D.3:

Nota: es necesario revisar la configuración de *ntop* si la sonda y él mismo están en subredes distintas y *ntop* solo atiende el tráfico de las máquinas locales. En tal caso, *ntop* no registraría la información recibida en los montajes anteriores.

Parámetro	**Descripción**
-V `pdu_version`	Versión de las PDUs que se retransmitirán.
-m `privacy_mask`	Permite poner una máscara para ocultar los hosts de la red.
-S `stat_interval`	Número de minutos del intervalo en el que *flow-fanout* imprimirá un mensaje en `stderr` indicando numero de flujos recibidos, paquetes procesados y flujos perdidos.
`localip/remoteip/port` `localip/remoteip/port` ...	En la primera tripleta: dirección IP local donde se esperan los flujos, IP de la que se esperan y puerto donde se esperan. En las demás tripletas: dirección IP del origen de los datos, IP del destino a donde se envían y su puerto. Esta segunda estructura se puede repetir tantas veces como recolectores haya a los que queramos enviar la información.

Cuadro D.6: Parámetros de *flow-fanout*

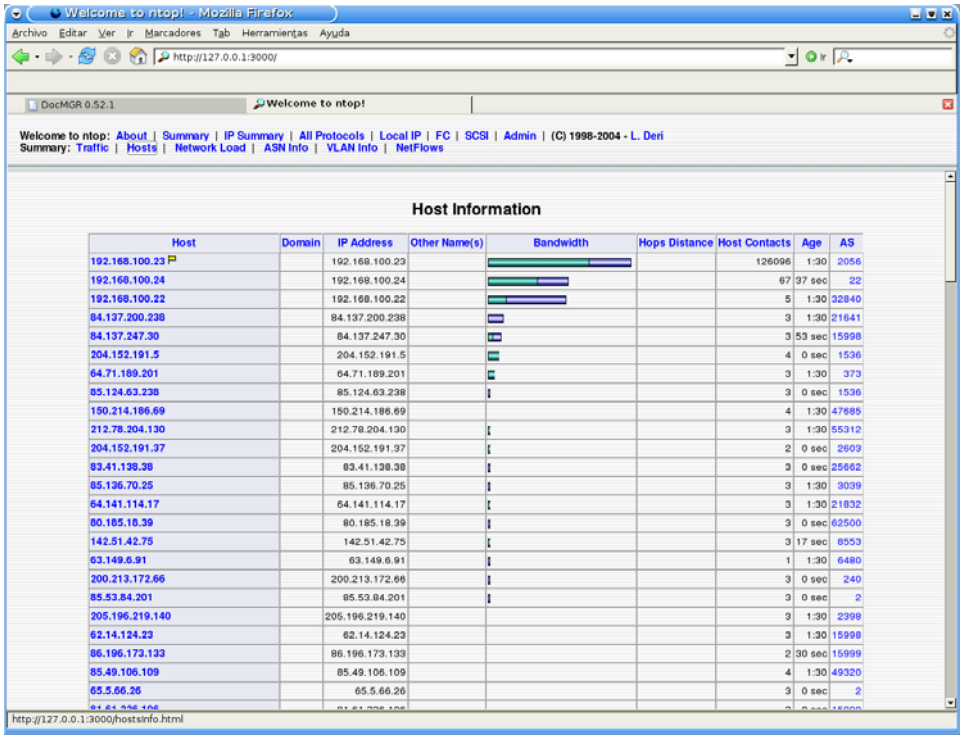

Figura D.3: *ntop* con la información ya exportada

D.1.3. Exportando a una base de datos *MySQL*

Estudiaremos en esta sección la posibilidad de exportar la información almacenada por el recolector a una base de datos *MySQL*. En caso de no disponer de un servidor de bases de datos *MySQL* ya instalado en el sistema se pueden consultar en la sección D.1.3.1, "*Instalando un servidor de base de datos MySQL*", los detalles para realizarlo.

El paquete ***flow-tools*** dispone de la utilidad *flow-export* para la exportación de la información almacenada por el recolector en diversos formatos, desde *ASCII* a bases de datos *MySQL* o *PostgreSQL*. Veamos cuáles son los parámetros de configuración de *flow-export*, indicados en el Cuadro D.7, para comprobar sus posibilidades. Más detalles sobre estos y otros parámetros pueden ser consultados en el manual de la aplicación [22].

Parámetro	**Descripción**
-f `format`	Especifica el formato de salida. Se dispone de: 0 *cflowd* 1 *pcap* 2 *ASCII CSV* 3 *MySQL* 4 *wire* 5 *PGSQ*
-m `mask`	Especifica qué campos serán exportados. Véase el Cuadro D.8.
-u `user:password:host:port:name:table`	Sólo para las exportaciones a *MySQL* o *PGSQ*, indican parámetros para el acceso al servidor de la base de datos. Especifica el usuario de la base de datos, su contraseña, la dirección del servidor, el puerto en el que atiende las solicitudes, el nombre de la base de datos y la tabla en la que se desea almacenar la información.

Cuadro D.7: Parámetros de *flow-export*

La máscara indicará que campos de la información almacenada en *NetFlow* queremos exportar hacia la salida. Para las exportaciones a bases de datos mostramos en el Cuadro D.8 la información de las máscaras, su relación con la información transportada en el protocolo *NetFlow* y el nombre del campo con el que *flow-export* intentaría exportarlos hacia la base de datos.

Máscara	**Valor numérico**	**Campo en *NetFlow* v7**	**Nombre de exportación**	**Notas**
UNIX_SECS	0x000000001	unix_secs	unix_secs	
UNIX_NSECS	0x000000002	unix_nsecs	unix_nsecs	
SYSUPTIME	0x000000004	SysUptime	sysuptime	
EXADDR	0x000000008		exaddr	IP del exportador del flujo.
DFLOWS	0x000000010		dflows	Número de flujos (agregaciones).
DPKTS	0x000000020	dPkts	dpkts	
DOCTETS	0x000000040	dOctets	doctets	
FIRST	0x000000080	First	first	
LAST	0x000000100	Last	last	
ENGINE_TYPE	0x000000200		engine_type	
ENGINE_ID	0x000000400		engine_id	
SRCADDR	0x000001000	srcaddr	srcaddr	
DSTADDR	0x000002000	dstaddr	dstaddr	
SRC_PREFIX	0x000004000			Aparece en el manual, pero no es exportable.
DST_PREFIX	0x000008000			Aparece en el manual, pero no es exportable.
NEXTHOP	0x000010000	nexthop	nexthop	Siempre vale 'cero'.
INPUT	0x000020000	input	input	Siempre vale 'cero'.
OUTPUT	0x000040000	output	output	
SRCPORT	0x000080000	srcport	srcport	
DSTPORT	0x000100000	dstport	dstport	
PROT	0x000200000	prot	prot	
TOS	0x000400000	tos	tos	
TCP_FLAGS	0x000800000	tcp_flags	tcp_flags	Siempre vale 'cero'.
SRC_MASK	0x001000000	src_mask	src_mask	Siempre vale 'cero'.
DST_MASK	0x002000000	dst_mask	dst_mask	Siempre vale 'cero'.
SRC_AS	0x004000000	src_as	src_as	Siempre vale 'cero'.
DST_AS	0x008000000	dst_as	dst_as	Siempre vale 'cero'.
IN_ENCAPS	0x010000000		in_encaps	Sólo *NetFlow* version 6. Tamaño de la encapsulación (entrada).
OUT_ENCAPS	0x020000000		out_encaps	Sólo *NetFlow* version 6. Tamaño de la encapsulación (salida).
PEER_NEXTHOP	0x040000000		peer_nexthop	
ROUTER_SC	0x080000000	router_sc	router_sc	
EXTRA_PKTS	0x100000000		extra_pkts	
MARKED_TOS	0x200000000		marked_tos	

Cuadro D.8: Relación de la exportación de *flow-export* con los campos de *NetFlow* V7

Puede verse que los campos del protocolo *NetFlow* que faltan en esta lista son: *version*, *count*, *flow-sequence*, *reserved*, *flags (x2)*. Estos campos coinciden en que no tienen significación fuera del envío de la información de la sonda al recolector, es decir, son información sobre el propio protocolo o sobre el envío de la información y ya no tienen sentido fuera de ese contexto. Así mismo, *flow-export* puede introducir información adicional a *NetFlow*, así como lo hacía *flow-print* (aunque no lo resaltamos en el capítulo D.1.1, "*Presentación mediante consola*") como la dirección de la sonda (*exaddr*).

Con esta información y podemos escribir la estructura de una base de datos *MySQL* [62, 63] que será a la cual enviaremos la información capturada. Para la siguiente estructura hemos omitido los campos que *NetFlow* siempre transportará con valor 'cero':

```
CREATE TABLE `tabla1` (
`sysuptime` int(15) NOT NULL default '0',
`srcaddr` varchar(15) NOT NULL default '',
`srcport` smallint(6) NOT NULL default '0',
`dstaddr` varchar(15) NOT NULL default '',
`dstport` smallint(6) NOT NULL default '0',
`prot` smallint(2) NOT NULL default '0',
`tos` smallint(6) NOT NULL default '0',
`unix_secs` int(15) NOT NULL default '0',
`unix_nsecs` int(15) NOT NULL default '0',
`dPkts` smallint(10) NOT NULL default '0',
`dOctets` int(15) NOT NULL default '0',
`first` int(15) NOT NULL default '0',
`last` int(15) NOT NULL default '0',
) TYPE=MyISAM;
```

Para mostrar como se exporta la información a una base de datos *MySQL* llamada *flows* con esta estructura se utilizarían indistintamente los siguientes comandos, donde el usuario de la base de datos sería *flow* y no tendría contraseña:

```
#> flow-cat /var/flow/eth0 | flow-export -f3 -mUNIX_SECS,UNIX_NSECS,SYSUPTIME,
     DPKTS,DOCTETS,FIRST,LAST,SRCADDR,DSTADDR,SRCPORT,DSTPORT,PROT,TOS -u flow::
     localhost:3306:flows:tabla1

#> flow-cat /var/flow/eth0 | flow-export -f3 -m0x7831EF -u flow::localhost:3306:
     flows:tabla1
```

Dicho usuario *flow* podría crearse con los siguientes comandos de *SQL*:

```
INSERT INTO user (host,user,password) VALUES('localhost','flow',PASSWORD(''));
GRANT SELECT , INSERT , UPDATE , DELETE ON `flows`.* TO 'flow'@'localhost';
FLUSH PRIVILEGES;
```

Por último, al basarse la exportación que *flow-export* realiza en una adquisición de datos a través de *flow-cat*, es posible realizar el mismo tratamiento de los datos que se hacía hacia *ntop* como se indicó en la sección D.1.2, "*Exportando a ntop*".

D.1.3.1. Instalando un servidor de base de datos *MySQL*

Cubrimos en este punto la instalación y configuración de un servidor de base de datos tipo *MySQL* para nuestra distribución **Debian Sarge**.

En primer lugar obtendremos los paquetes necesarios:

```
#> apt-get install mysql-server-4.1
```

Esto nos instalará el paquete ***mysql-server-4.1*** en el sistema junto a una extensa lista de paquetes que son dependencias del mismo.

El servidor *MySQL* se instala pro defecto sin contraseña para el usuario `root`. Para establecer una contraseña para dicho usuario podemos ejecutar la siguiente orden:

```
#> mysqladmin -u root password NUEVO_PASSWORD
```

donde «NUEVO_PASSWORD» será la nueva contraseña que queremos establecer. «password» sería en circunstancias normales la contraseña actual del usuario `root` de *MySQL*, pero que en este caso no está establecida y acepta este valor. Podemos comprobar que hemos establecido la contraseña deseada como sigue:

```
#> mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8 to server version: 4.0.24_Debian-10sarge2-log

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> exit
Bye

#>
```

La configuración del servidor reside en `/etc/mysql/my.cnf`. Dicho archivo está bien comentado y es autoexplicativo. Nosotros, en principio, no necesitamos realizar ninguna modificación más a dicha configuración.

D.1.4. Exportando a bases de datos *Round Robin*

Estudiaremos en esta sección la posibilidad de exportar la información almacenada por el recolector a una base de datos *Round Robin*. Las bases de datos *Round Robin* o *RRD* tienen la ventaja, como se comentó en la sección 7.1.3, "*Exportando a una base de datos*", de tener tamaño fijo por lo que tendremos menos problemas de almacenamiento en disco que con las bases de datos *MySQL*.

Existen una serie de herramientas que podemos configurar para obtener gráficas del tráfico que atraviesa nuestros sistemas remotos una vez que hemos recibido la información mediante *NetFlow* procedente desde nuestra sonda *fprobe-ng* y se ha procedido a su captura con *flow-capture*. Esta herramienta es *FlowScan* [58, 64, 65], a la que añadiremos dos complementos, *CUFlow* y *CUGrapher* [66], para poder fabricar las gráficas de uso de la red. Todas estas herramientas se presentan en paquetes para **Debian Sarge** en versión *testing*, y en los siguientes puntos veremos cómo proceder a la instalación y configuración de estas utilidades.

D.1.4.1. Obteniendo los paquetes

Para empezar necesitaremos cambiar temporalmente nuestra distribución **Debian Sarge** de *stable* a *testing* para lo que será necesario modificar el archivo `/etc/apt/sources.list`:

`/etc/apt/sources.list` para versión *testing*

```
# Repositorios para version "stable"
#deb http://ftp.es.debian.org/debian/ stable main
#deb-src http://ftp.es.debian.org/debian/ stable main

#deb http://ftp.funet.fi/pub/linux/mirrors/debian/ stable main
#deb-src http://ftp.funet.fi/pub/linux/mirrors/debian/ stable main

#deb http://security.debian.org/ stable/updates main

# Repositorios para version "testing"
deb http://ftp.es.debian.org/debian/ testing main
deb-src http://ftp.es.debian.org/debian/ testing main
```

Hecho esto, podemos pasar a usar *apt* para obtener los paquetes:

```
#> apt-get update
...

#> apt-get install flowscan flowscan-cuflow flowscan-cugrahper
```

Los contenidos de los paquetes se podrán consultar con las siguientes instrucciones:

```
#> dpkg -L flowscan
```

```
#> dpkg -L flowscan-cuflow
```

```
#> dpkg -L flowscan-cugrapher
```

D.1.4.2. Configurando *FlowScan*

Las *scripts* en Perl que forman *FlowScan* se almacenarán en `/usr/share/perl5` y las configuraciones de las herramientas en `/etc/flowscan`. Comenzaremos viendo la configuración de *FlowScan* ubicada en `/etc/flowscan/flowscan.cf`:

1. La directiva `FlowFileGlob` indica donde está el directorio que contiene los archivos a procesar. *FlowScan* no puede leer la estructura de directorios creada por *flow-capture* así que es necesario cambiar el parámetro `nesting_level` de la configuración de *flow-capture* al valor 0 para que no cree subdirectorios, como se indicó en el Cuadro A.4. Nosotros vamos a suponer que el directorio es `/var/flow/eth0`:

`/etc/flowscan/flowscan.cf` (I)

```
# flowscan Configuration Directives
    ##########################################

# FlowFileGlob (REQUIRED)
# use this glob (file pattern match) when looking for raw flow files to be
# processed, e.g.:
# FlowFileGlob /var/local/flows/flows.*:*[0-9]
# FlowFileGlob flows.*:*[0-9]
FlowFileGlob /var/flow/eth0/ft-v*[0-9]
```

Es importante decir que *FlowScan* una vez ha analizado un archivo lo elimina, a no ser que exista un directorio llamado `saved` en el directorio donde se encuentran los archivos (en este caso sería `/var/flow/eth0/saved`). Si

existe dicho directorio *FlowScan* moverá ahí los archivos ya procesados, y este directorio estará sujeto además al posible control espacio en disco (de número de archivos o de tamaño) establecido en la configuración del recolector (ver el Cuadro A.4 en la sección A.3.2, "*Configuración de flow-capture*" para más información acerca del control de espacio en disco). Esto es muy útil para la depuración de errores ya que evita que perdamos los flujos capturados.

Así mismo, si queremos que procese más de un directorio lo que debemos hacer es escribir la lista de directorios a utilizar dentro de la directiva `FlowFileGlob`.

`/etc/flowscan/flowscan.cf` (Ejemplo para varios directorios)

```
# flowscan Configuration Directives
    ##########################################

# FlowFileGlob (REQUIRED)
# use this glob (file pattern match) when looking for raw flow files to be
# processed, e.g.:
# FlowFileGlob /var/local/flows/flows.*:*[0-9]
# FlowFileGlob flows.*:*[0-9]
FlowFileGlob /var/flow/eth0/ft-v*[0-9] /var/flow/eth1/ft-v*[0-9]
```

2. Lo siguiente es indicar a *FlowScan* que módulo se usará para el estudio de los archivos con la directiva `ReportClasses`. Nosotros usaremos `CUFLow`, que es el módulo que hemos añadido con los paquetes ***flowscan-cuflow*** y ***flowscan-cugrahper***:

`/etc/flowscan/flowscan.cf` (II)

```
# ReportClasses (REQUIRED)
# a comma-seperated list of FlowScan report classes, e.g.:
# ReportClasses CampusIO
# ReportClasses SubNetIO
ReportClasses CUFlow
```

Al igual que ocurre con la directiva `FlowFileGlob`, si deseamos que se trate la información capturada por varios módulos bastará escribir la lista de módulos deseados en la directiva `ReportClasses`. No obstante, estos módulos deberán ser compatibles entre sí.

3. Lo siguiente es indicar a *FlowScan* cuanto tiempo deseamos que pase entre dos estudios consecutivos del directorio que contiene los archivos, mediante la directiva `WaitSeconds`:

`/etc/flowscan/flowscan.cf` (III)

```

# WaitSeconds (OPTIONAL)
# This should be <= the "-s" value passed on the command-line to cflowd, e
    .g.:
# WaitSeconds 300
WaitSeconds 30
```

En períodos de prueba, para agilizar el tráfico, es conveniente reconfigurar *flow-capture* para que grabe los archivos a un ritmo mayor de uno cada 15 minutos (su configuración por defecto) mediante el parámetro `rotations` que se vió en el Cuadro A.4. Un valor de `1439` hará que se creen cada minuto (el ritmo más rápido posible).

4. Por último, indicar si queremos que *FlowScan* nos muestre una salida detallada de su comportamiento:

`/etc/flowscan/flowscan.cf` (y IV)

```
20
    # Verbose (OPTIONAL, non-zero = true)
    Verbose 1
```

Darle valor «1» a `Verbose` mostrará la salida detallada. Es necesario advertir que este valor es muy práctico para la depuración y comprobación del buen funcionamiento de la utilidad, pero su uso, sobre todo si se redirige dicha salida a un fichero, generará archivos de gran tamaño en nuestro sistema.

Así mismo hemos querido desarrollar una *script* que permita el manejo de *FlowScan* como si de otro servicio o demonio del sistema se tratase. A continuación presentamos dicha *script*:

`/etc/init.d/flowscan` (*script* de arranque)

```
    #!/bin/sh
    # description: Start FlowScan


 5  start(){
        FLOWSCAN_PID=`ps axjf | grep -v axjf | grep -v grep | grep "/usr/bin/
            flowscan" | sort -k 9 | tail -n 1 | awk '{print $2}'`

        if [ "$FLOWSCAN_PID" == "" ]
        then
10          /usr/bin/flowscan >>/var/log/flowscan 2>&1 </dev/null & >/dev/
                null
            #touch /var/lock/flowscan.1
            sleep 1
            FLOWSCAN_PID=`ps axjf | grep -v axjf | grep -v grep | grep "/usr
                /bin/flowscan" | sort -k 9 | tail -n 1 | awk '{print $2}'`
            echo "Starting flowscan service... "
15
            if [ "$FLOWSCAN_PID" == "" ]
            then
                echo "ERROR: flowscan is not running."
            else
20              echo "Started with PID "$FLOWSCAN_PID
            fi

        else
            echo "ERROR: flowscan is already running."
25      fi
        echo
    }

    stop(){
30      FLOWSCAN_PID=`ps axjf | grep -v axjf | grep -v grep | grep "/usr/bin/
            flowscan" | sort -k 9 | tail -n 1 | awk '{print $2}'`

        if [ "$FLOWSCAN_PID" == "" ]
        then
            echo "ERROR: flowscan is not running."
35      else
            kill $FLOWSCAN_PID
            #rm -f /var/lock/flowscan.1
            echo "Stopping flowscan service... Found PID "$FLOWSCAN_PID",
                killing it"
        fi
40      echo
    }
```

```
    case "$1" in
    start)
45          start

    ;;
    stop)
            stop
50  ;;

    restart)
            stop
            start
55  ;;
    *)
            echo "Usage: $0 { start | stop | restart}"
    ;;
    esac
60  exit 0
```

La *script* es simple: dado que *FlowScan* no genera archivos PID durante su ejecución, hemos tenido que recurrir a comprobar si el servicio está activo mediante *ps*. La *script* permite las habituales llamadas *start*, *stop* y *restart* de los demonios del sistema. La emplazaremos en el archivo `/etc/init.d/flowscan`, y podremos hacer que se inicie en el arranque del sistema y se detenga en su apagado con la siguiente orden:

```
#> update-rc.d flowscan defaults 25
 Adding system startup for /etc/init.d/flowscan ...
   /etc/rc0.d/K25flowscan -> ../init.d/flowscan
   /etc/rc1.d/K25flowscan -> ../init.d/flowscan
   /etc/rc6.d/K25flowscan -> ../init.d/flowscan
   /etc/rc2.d/S25flowscan -> ../init.d/flowscan
   /etc/rc3.d/S25flowscan -> ../init.d/flowscan
   /etc/rc4.d/S25flowscan -> ../init.d/flowscan
   /etc/rc5.d/S25flowscan -> ../init.d/flowscan
```

Si deseamos eliminarlo del arranque y detención del sistema basta ejecutar:

```
#> update-rc.d -f flowscan remove
update-rc.d: /etc/init.d/flowscan exists during rc.d purge (continuing)
 Removing any system startup links for /etc/init.d/flowscan ...
   /etc/rc0.d/K25flowscan
   /etc/rc1.d/K25flowscan
   /etc/rc2.d/S25flowscan
   /etc/rc3.d/S25flowscan
   /etc/rc4.d/S25flowscan
   /etc/rc5.d/S25flowscan
   /etc/rc6.d/K25flowscan
```

Por último, en dicha *script* se indica al programa que dirija su salida hacia el archivo ubicado en `/var/log/flowscan` en caso de que estemos haciendo uso de la salida detallada.

D.1.4.3. Configurando *CUFlow*

Una vez configurado *FlowScan* configuraremos *CUFlow*, el módulo que usaremos para analizar los archivos. Su configuración reside en `/etc/flowscan/CUFlow.cf`:

1. Lo primero que *CUFlow* nos permite es indicar las subredes que componen nuestra red:

`/etc/flowscan/CUFlow.cf` (I)

```
    # These are the subnets in our network
```

```
    # These are used only to determine whether a packet is inbound our
    # outbound
    # Subnet 10.0.0.0/16
    Subnet 172.26.0.0/23
```

2. Con la directiva `Network` indicamos las redes de las que queremos tener luego información separada:

`/etc/flowscan/CUFlow.cf` (II)

```
    # These are networks we are particularly interested in, and want to
    # get separate rrd's for their aggregate traffic
    # Network 10.0.1.0/24 routers
    Network 172.26.0.0/24 backbone
    Network 172.26.1.0/24 routers
```

3. La siguiente directiva es la que indica a *CUFlow* el directorio donde se almacenarán los archivos *RRD* que contendrán sus resultados:

`/etc/flowscan/CUFlow.cf` (III)

```
    # Where to put the rrd's
    # Make sure this is the same as $rrddir in CUGrapher.pl
    # OutputDir /cflow/reports/rrds
    OutputDir /var/CUFlow/rrds
```

   Este directorio también hay que introducirlo dentro del archivo `/etc/flowscan/CUGrapher.cf`, como se indicará posteriormente. No lo indica la documentación, pero se ha encontrado que es necesario crear también un directorio que cuelgue del padre del especificado con `OutputDir` y que se llame `scoreboard` para el correcto funcionamiento del programa. En este ejemplo concreto habrá que crear entonces los directorios `/var/CUFlow/rrds` y `/var/CUFlow/scoreboard`.

4. Las directivas `Scoreboard` y `AggregateScore` permiten realizar una salida en formato HTML de un estudio de los usuarios más consumidores de ancho de banda. La primera creará el estudio de los más consumidores del último período estudiado y la segunda realizará un histórico. Las salidas deberán ser emplazadas dentro de un dirctorio válido para ser servido por el servidor *Web* instalado en el sistema:

`/etc/flowscan/CUFlow.cf` (IV)

```

    # Keep top N lists
    # Show the top ten talkers, storing reports in /cflow/flows/reports
    # and keeping the current report in /etc/httpd/data/reports/topten.html
    # Scoreboard 10 /cflow/reports/scoreboard /var/www/html/topten.html
    Scoreboard 10 /var/CUFlow/scoreboard /var/www/topten.html

    # Same, but build an over-time average top N list
    # AggregateScore 10 /cflow/reports/scoreboard/agg.dat /var/www/html/
        overall.html
    AggregateScore 10 /var/CUFlow/scoreboard/agg.dat /var/www/overall.html
```

5. El parámetro `Router` permite diferenciar a los exportadores, nuestras sondas, de los archivos que están siendo analizados:

`/etc/flowscan/CUFlow.cf` (y V)

```
30
    # Our two netflow exporters. Produce service and protocol reports for the
    # total, and each of these.
    #Router 10.0.1.1 router1
    #Router 10.0.1.2 router2
35  Router 172.26.0.1 routerSalida
```

6. Se incluyen más parámetros en la configuración que ya no necesitamos configurar, pero cuyo uso puede ser interesante para establecer definir los protocolos que se analizarán, los servicios, los *AS* que diferenciamos, etc...

Configurado ya *CUFlow*, *FlowScan* es ya capaz de analizar los archivos que hemos generado anteriormente con *flow-capture* y nos generaría las bases de datos *RRD* además de las salidas de `Scoreboard` y `AggregateScore`, como se muestra en la Figura D.4. Para ello tan sólo tenemos que ejecutar la herramienta *FlowScan*:

```
#> flowscan
...
```

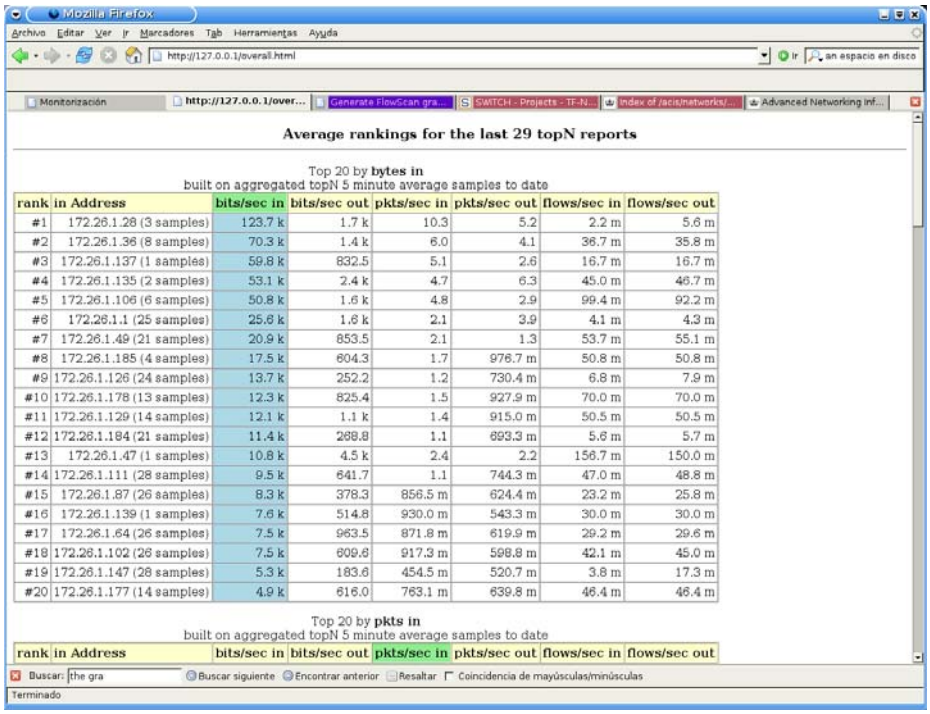

Figura D.4: `Scoreboard` de *CUFlow*

D.1.4.4. Configurando *CUGrapher*

Ahora tan solo nos falta configurar la parte gráfica de las herramientas, es decir, la *script* en Perl *CUGrapher* que viene incluída dentro del paquete ***flowscan-cugrapher***. Para ello sólo hay que hacer algunas modificaciones a su

archivo de configuracion, `/etc/flowscan/CUGrapher.cf`, para hacerla funcionar:

1. Es necesario indicarle a la *script* donde se encuentran los archivos *RRD*:

`/etc/flowscan/CUGrapher.cf` (I)

```
# Configuration file for CUGrapger.pl.
#

# Where to find the rrd's.  You must set this.
# Make sure it is the same as OutputDir in CUFlow.cf.
#
# Eg:       OutputDir /cflow/reports/rrds
# Default: not set
OutputDir /var/CUFlow/rrds
```

   Esta ruta será la misma que la que se especificó en la directiva `OutputDir` de *CUFlow*.

2. Los demás parámetros son autoexplicativos y fácilmente configurables. Pueden presentar un mayor interés las directivas `Organization` y `Title` para la representación de nuestras gráficas.

`/etc/flowscan/CUGrapher.cf` (y II)

```

# Organisation name - appears in the graph title.
#
# Eg:       Organization Columbia University Campus
# Default: (blank)
Organization Escuela Superior de Ingenieros - CUFlow

# Default number of hours to go back.
#
# Eg:       Hours 24
# Default: Hours 48

# Default width of graph in pixels.
#
# Eg:       Width 800
# Default: Width 640

# Default height of graph in pixels.
#
# Eg:       Height 480
# Default: Height 320

# Default image type.  Can be png or gif.
#
# Eg:       ImageType gif
# Default: ImageType png

# Default graph title.
#
# Eg:       Title My Graph
# Default: Title Well Known Protocols/Services

#
# Specify a graph to be displayed on startup, when no query
# has been entered.  To display multiple graphs supply
# multiple "DefaultGraph" lines.  To generate the string
# following the "DefaultGraph":
#  1.  Use the web page to generate the graph you want.
#  2.  Copy the query part of the URL displayed by your browser
#      (ie everything part the '?').
#  3.  Remove the ';showmenu=1' from the query string copied.
#
```

```
    # Eg:       DefaultGraph report=bits;hours=48;imageType=png;width=640;
        height=320;duration=;router=all;all_all_services=1;legend=1;title=My
        %20Graph
    # Default: not set
55
    # The path to the AggregateScore web page build by CUFlow.  If
    # non-blank a like to AggregateScore web page will be displayed.
    # If supplied this must be the same as the file name given to
    # the AggregateScore setting in CUFlow.cf.
60  #
    # Eg:       AggregateScore /var/local/netflow/cuflow/agg10.html
    # Default: (not set)

    # The path to the Scoreboard web page built by CUFlow.  If
65  # non-blank a link to the Scoreboard web page will be
    # displayed.  If supplied this must be the same as the file
    # name given to the Scoreboard setting in CUFlow.cf.
    #
    # Eg:       Scoreboard /var/local/netflow/cuflow/top10.html
70  # Default: (not set)
```

Tras ello, solo nos queda ejecutar la *script* `CUGrapher.cgi` que está emplazada dentro de los directorios de `cgi-bin` de nuestro servidor *Web*. Para ello tendríamos que introducir en nuestro navegador *Web* una dirección como la siguiente: http://127.0.0.1/cgi-bin/CUGrahper.cgi, y obtendríamos un resultado como el mostrado en la Figura D.5:

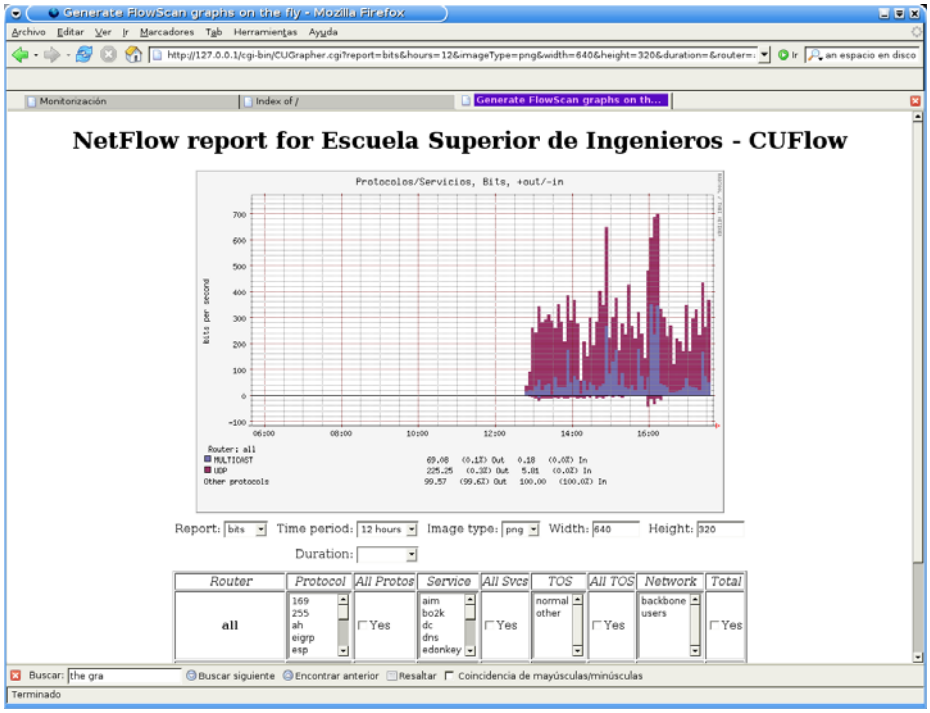

Figura D.5: Presentación de *CUGrapher*

D.1.4.5. Instalando y configurando *FlowMonitor*

Además de los módulos anteriores también se ha estudiado otro complemento más de *FlowScan* llamado *FlowMonitor* [67] que permite realizar otro tipo de informes de consumo identificando a los usuarios y presentándolos diferenciados. Este módulo nos permite establecer unos umbrales que al ser sobrepasados harán que los usuarios que los sobrepasen se nos presenten en una lista. Sin embargo, a pesar de ser un complemento de la utilidad *FlowScan*, *FlowMonitor* necesita de una base de datos *MySQL* para su funcionamiento pero hemos querido tratarla aquí dada su dependencia de la utilidad *FlowScan*. ***FlowMonitor*** no se encuentra disponible en *paquete Debian*, pero podemos obtenerlo de la siguiente forma:

```
#> cd /usr/src

#> wget http://www.columbia.edu/acis/networks/advanced/FlowMonitor/FlowMonitor
    -1.2.tar.gz
...

#> tar -xvjf FlowMonitor-1.2.tar.gz
```

Esto nos dejará los siguientes archivos:

```
/usr/src/FlowMonitor-1.2/COPYING
/usr/src/FlowMonitor-1.2/FlowMonitor.cf
/usr/src/FlowMonitor-1.2/FlowMonitor.pm
/usr/src/FlowMonitor-1.2/README.txt
```

Para hacer el comportamiento de *FlowMonitor* similar al de los otros módulos de *FlowScan*, emplazaremos su archivo de configuración y su *script* en las mismas ubicaciones:

```
#> cp /usr/src/FlowMonitor-1.2/FlowMonitor.cf /etc/flowscan/FlowMonitor.cf

#> cp /usr/src/FlowMonitor-1.2/FlowMonitor.pm /usr/share/perl5/FlowMonitor.pm
```

Para conseguir esto, necesitamos cambiar la ubicación del archivo de configuración dentro de `FlowMonitor.pm` para que la localice correctamente:

`/usr/share/perl5/FlowMonitor.pm` (modificado)

```
    $ENV{'ORACLE_HOME'} = "/opt/oracle-9.2" unless defined $ENV{'ORACLE_HOME'};
260 #&parseConfig("${FindBin::Bin}/FlowMonitor.cf"); # Read our config file
    &parseConfig("/etc/${FindBin::Script}/FlowMonitor.cf");      # Read our config
        file
```

Así mismo, deberemos modificar la configuración de *FlowScan*, emplazada en `/etc/flowscan/flowscan.cf`, para que haga uso de este módulo:

`/etc/flowscan/flowscan.cf` (modificado)

```
 10 # ReportClasses (REQUIRED)
    # a comma-seperated list of FlowScan report classes, e.g.:
    # ReportClasses CampusIO
    # ReportClasses SubNetIO
    # ReportClasses CUFlow
 15 ReportClasses FlowMonitor
```

FlowMonitor y *CUFlow* no son compatibles entre sí.

FlowMonitor necesita una base de datos para almacenar su información temporal, y por defecto querría usar una base de datos *Oracle* para ello. Nosotros

cambiaremos la configuración para reutilizar la base de datos *MySQL* que ya utilizamos anteriormente en el punto D.1.3, "*Exportando a una base de datos MySQL*". Primero necesitaremos modificar el archivo `/usr/share/perl5/FlowMonitor.pm` para indicar el nombre del usuario de la base de datos y su clave, que en este caso será el usuario *flow* y no tendrá clave:

`/usr/share/perl5/FlowMonitor.pm` (modificado)

```
     my($SUBNETS);                  # A trie of internal subnets. IPs we
                                    # should keep data on.
     my(%ROUTERS);                  # A hash of ip's we should
                                    # police records from
225  my($INUSAGE);                  # A trie for recording the usage of IPs
                                    # we find in flow records for inbound data
     my($OUTUSAGE);                 # A trie for recording the usage of IPs
                                    # we find in flow records for inbound data
     my($IGNORE);                   # A trie of ips we should not list
230  my($QUOTA);                    # Bytes allowed in a given INTERVAL
     my($INTERVAL);                 # Time between resets of an IPs counter, in
                                    # units of seconds
     my(%VIOLATORS);                # Hash keyed by %age of policy, whose value
                                    # is a filename to write IPs to
235  my($DBNAME) = 'dbi:';          # Which DBI module to use
     my($DBUSER) = 'flow';             # What username to connect as
     my($DBPASS) = '';              # What password to connect with

     my($SYSDATE) = '';             # What to select to get the current time
240  my($DATEUNITS) = '';           # Number of seconds in the units $SYSDATE
                                    # math is done in. Ie, if $SYSDATE - 1 is
                                    # 1 day before sysdate, then DATEUNITS will
                                    # be 86400 (seconds per day)
```

Para crear el usuario de *MySQL* adecuado y sin contraseña podremos usar los mismos comandos de *SQL* vistos en la sección D.1.3, "*Exportando a una base de datos MySQL*".

Únicamente falta adaptar la configuración de *FlowMonitor* a nuestro sistema. El archivo de configuración se encuentra en `/etc/flowscan/FlowMonitor.cf`:

1. *FlowMonitor* nos permite primero establecer una serie de subredes y exportadores de *NetFlow*:

`/etc/flowscan/FlowMonitor.cf` (I)

```
     # Example

     #Subnet 160.39.0.0/16
     #Subnet 192.5.43.0/24
  5  #Subnet 67.99.58.192/30
     #Subnet 128.59.0.0/16
     #Subnet 156.111.0.0/16
     #Subnet 129.236.0.0/16
     #Subnet 156.145.0.0/16
 10  #Subnet 207.10.136.0/21
     #Subnet 209.2.47.0/24
     #Subnet 209.2.48.0/22
     #Subnet 209.2.185.0/24
     #Subnet 209.2.208.0/20
 15  #Subnet 209.2.224.0/20
     Subnet 172.26.0.0/24
     Subnet 172.26.1.0/24

     #Router 128.59.1.4
 20  Router 172.26.0.1
```

2. La directiva `Carry-Forward` permite tener un mejor seguimiento del consumo de los usuarios. Si tenemos el limite establecido a 100 MegaBytes por hora y un usuario consume 120 MegaBytes, con esta directiva activada la siguiente hora comenzará con 20 MegaBytes en su cuenta en lugar de con 0:

`/etc/flowscan/FlowMonitor.cf` (II)

```
20  Router 172.26.0.1

    Carry-Forward
```

3. Lo siguiente es la definición de la política (`Policy`) del límite de consumo. Soporta tres métodos distintos de contabilidad según se especifique `Inbound` (donde sólo se contabiliza el tráfico que tiene como destino un elemento contenido dentro de una definición previa de `Subnet`), `Outbound` (donde se contabilizará el tráfico originado por un elemento definido en una `Subnet`) o `Both` (contabilizará el tráfico con origen o destino comprendido en las definiciones de `Subnet`):

`/etc/flowscan/FlowMonitor.cf` (III)

```

    # 100 MB / 1 hours
25  Policy Outbound 100000000 3600
    # 60 MB / 1 hours
    #Policy Outbound 60000000 3600
    # 200 MB / 1 hours
    #Policy 200000000 3600
30  # 4 GB / 24 hours
    #Policy 4320000000 86400
    # 8 GB / 24 hours
    #Policy 8640000000 86400
    # 2 MB / 10 minutes
35  #Policy 2000000 600
```

4. Con la directiva `DBName` especificamos la base de datos a utilizar. En lugar de usar una base de datos Oracle utilizaremos una *MySQL*, concretamente una llamada *flows*:

`/etc/flowscan/FlowMonitor.cf` (IV)

```
35  #Policy 2000000 600

    #DBName Oracle oradb
    DBName mysql flows
```

FlowMonitor no es especialmente flexible en estos aspectos. Dentro de la base de datos ***flows*** necesariamente debe existir una tabla llamada *iplogs* con el siguiente formato:

```
CREATE TABLE `iplogs` (
        `ipaddr` VARCHAR( 16 ) NOT NULL ,
        `bytes` INT( 16 ) NOT NULL ,
        `starttime` DATETIME NOT NULL ,
        UNIQUE (`ipaddr`)
) TYPE = MyISAM ;
```

5. La directiva `Violators` es utilizada para marcar a los usuarios que sobrepasan un cierto nivel dentro de la política establecida. El formato es fácil de entender con la siguiente configuración:

`/etc/flowscan/FlowMonitor.cf` (V)

```
    DBName mysql flows

40  # Violators 0 /var/www/html/violators-0.txt

    # Every user gets logged to 0percenters
    Violators 0 /var/www/0percenters
    # People at 50% over policy get logged to 50percenters
45  Violators 50 /var/www/50percenters
    # People at 100% over policy get logged to 100percenters
    Violators 100 /var/www/100percenters
    # People at 200% or more over policy go to losers
    Violators 200 /var/www/losers
```

Un usuario solo aparecerá en el límite más alto que haya sobrepasado, es decir, si un usuario sobrepasa el 100 % de la política solo aparecerá en `100percenters` en el ejemplo anterior.

6. Con la directiva `Logfile` podemos llevar un registro de qué usuario estuvo en qué archivo a qué hora:

`/etc/flowscan/FlowMonitor.cf` (VI)

```
    Violators 200 /var/www/losers
50
    #Logfile /var/www/html/violator-history.txt
    Logfile /var/www/violator-history.txt
```

7. Por último, las directivas `Ignore` e `IgnoreList` permiten, respectivamente, establecer una serie de bloques de red (especificados en notación CIDR) a los que no se contemplará para la lista de `Violators` y exportar dicha lista a un archivo de texto.

`/etc/flowscan/FlowMonitor.cf` (VII)

```
    IgnoreList /var/www/html/ignorelist.txt

55  # Test ignore
    Ignore 128.59.1.1/32

    # Ignore Machine room
    Ignore 128.59.59.0/24
```

Ahora, al ejecutar la herramienta *FlowScan* se crearán los archivos de violaciones en los lugares que hemos especificado al proceder al análisis de los archivos que contienen las capturas de *NetFlow*.

D.2. Tratamiento de la información de los registros

D.2.1. Visualización en consola y página *Web*: *ccze*

ccze es un simple visualizador de registros. *ccze* está preconfigurado para mostrar una serie de registros de distintos programas, como *fetchmail*, *proftpd*, *squid*, o el que a nosotros nos interesa, *syslog-ng* [68].

ccze no hace ningún tipo de análisis o resumen de los contenidos del archivo del registro sino que lo muestra tal y como es pero aplicándole una *colorización* por defecto. Esto tiene la utilidad que, junto a su capacidad de volcar sus resultados a una consola o terminal o hacia una página *Web* que se actualizarán de forma automática, permitirán a un administrador del sistema estar al tanto de los contenidos de los registros sin tener que esperar al resumen que otros programas hagan de forma periódica.

ccze se proporciona como *paquete Debian* en su versión 0.2.1-1, y su instalación es rápida y sencilla:

```
#> apt-get install ccze
```

Se podrán consultar los contenidos del paquete de la siguiente forma:

```
#> dpkg -L ccze
```

Como hemos comentado, *ccze* está en su mayor parte preconfigurado y no permite realizar modificaciones importantes sobre su funcionamiento. El único archivo de configuración modificable es `/etc/cczerc`. Dicho archivo sólo nos permite modificar la colorización que cada uno de los campos del registro sufre una vez reconocido. No nos deja, por ejemplo, realizar una colorización en función del *host* que emite el mensaje, una característica que sería útil y que echamos en falta. Lo que sí nos permite *ccze* es alguna opción en su ejecución, como se muestra en el Cuadro D.9. Más información acerca de sus parámetros puede encontrarse en su manual [69].

Parámetro	**Descripción**
-A	Salida en formato *raw ANSI*
-h	Salida en formato HTML

Cuadro D.9: Parámetros de *ccze*

Su puesta en funcionamiento será la siguiente: configuraremos a *syslog-ng* para que una nueva salida de registro sea el propio *ccze*, volcando sus resultados a distintas salidas:

`/etc/syslog-ng/syslog-ng.cfg` (líneas añadidas)

```
destination df_ccze {
        program("ccze > /dev/tty8");
};

destination df_cczeweb {
```

```
            program("ccze -h > /var/www/ccze.html");
    };

    log {
10          source(s_all);
            filter(f_syslog);
            destination(df_ccze);
    };

15  log {
            source(s_all);
            filter(f_syslog);
            destination(df_cczeweb);
    };
```

Con esta configuración, hacemos que *syslog-ng* filtre los mensajes y los mande a *ccze* de dos formas distintas: la primera los mandará al terminal número 8 y la segunda a un archivo de formato HTML. Suponemos que la ruta `/var/www` está dentro de la ruta habilitada para el servidor *apache*. Se adjuntan a continuación dos imágenes en las que se puede ver el funcionamiento de *ccze* tanto en modo consola como exportando a página *Web*:

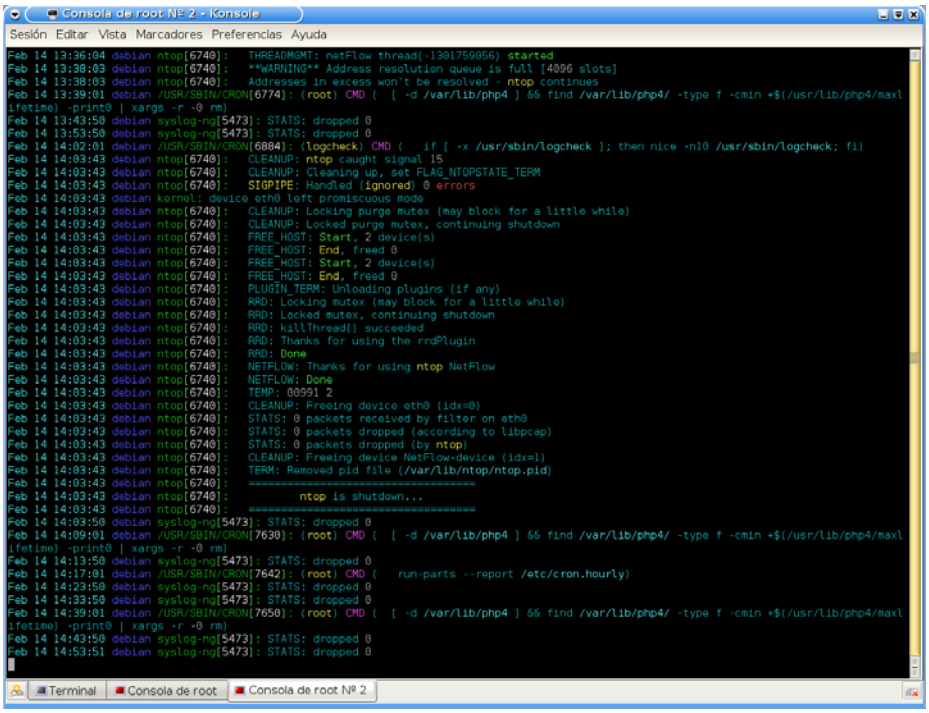

Figura D.6: Uso de *ccze* mediante consola

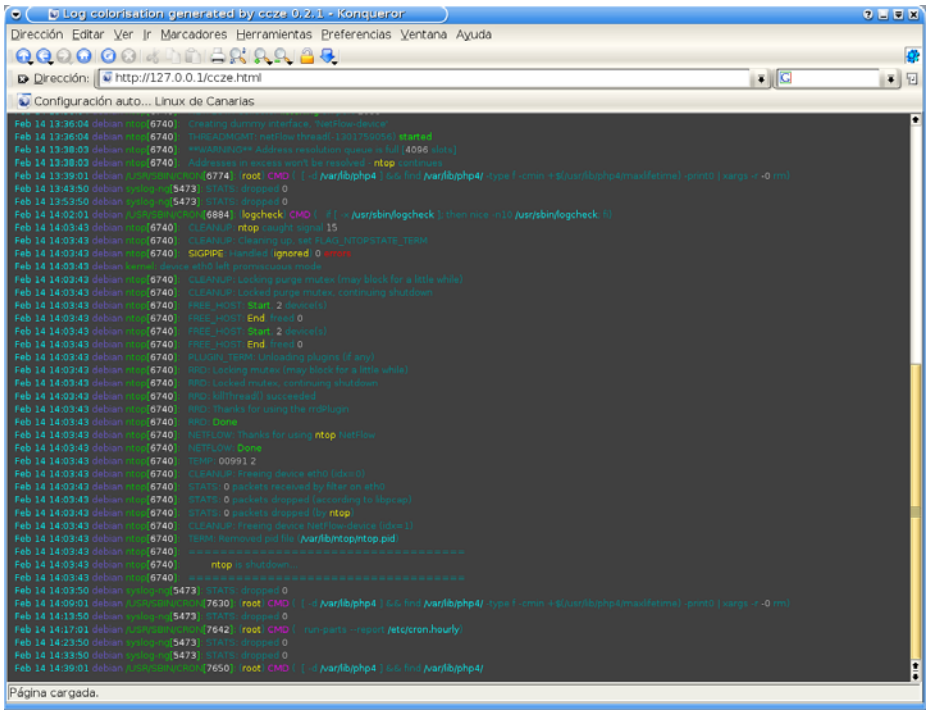

Figura D.7: Uso de *ccze* mediante página *Web*