User Documentation nfdump & NfSen

1 NFDUMP

This is the combined documentation of nfdump & NfSen. Both tools are distributed under the BSD license and can be downloaded at

nfdump	http://sourceforge.net/projects/nfdump/
nfsen	http://sourceforge.net/projects/nfsen/

This documentation describes nfdump tool v1.5 and NfSen v1.2.3.

1.1 NFDUMP tools overview

All tools support netflow v5, v7 and v9.

nfcapd - netflow capture daemon.
Reads the netflow data from the network and stores the data into files. Automatically rotate files every n minutes. (typically every 5 min) nfcapd reads netflow v5, v7 and v9 flows transparently. You need one nfcapd process for each netflow stream.

nfdump - netflow dump.
Reads the netflow data from the files stored by nfcapd. It's syntax is similar to tcpdump. If you like tcpdump you will like nfdump. Nfdump displays netflow data and can create lots of top N statistics of flows IP addresses, ports etc ordered by whatever order you like.

nfprofile - netflow profiler.
Reads the netflow data from the files stored by nfcapd. Filters the netflow data according to the specified filter sets (profiles) and stores the filtered data into files for later use. Mostly used by NfSen.

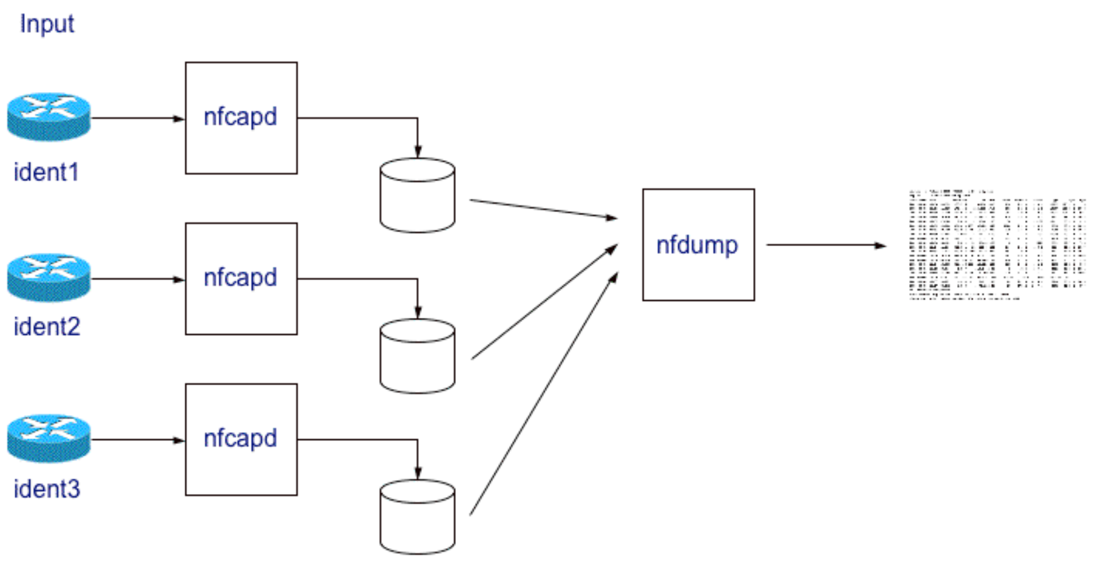
nfreplay - netflow replay
Reads the netflow data from the files stored by nfcapd and sends it over the network to another host.

nfclean.pl - cleanup old data
Sample script to cleanup old data. You may run this script every hour or so.

ft2nfdump – Optional binary: Reads and converts flow-tools data.
Reads flow-tools data from files or from stdin in a chain of flow-tools commands and converts the data into nfdump format to be processed by nfdump.

1.2 Principle of Operation:

The goal of the design is to be able to analyze netflow data from the past as well as to track interesting traffic patterns continuously. The amount of time back in the past is limited only by the disk space available for all the netflow data. The tools are optimized for speed for efficient filtering. The filter rules look familiar to the syntax of tcpdump (pcap like).



All data is stored to disk, before analyzing. This separates the process of storing and analyzing the data.

The data is organized in a time based fashion. Every n minutes - typically 5 min - nfcapd rotates and renames the output file with the time stamp **nfcapd.YYYYMMddhhmm** of the interval e.g. **nfcapd.200407110845** contains data from July 11th 2004 08:45 onward. Based on a 5min time interval, this results in 288 files per day.

Analyzing the data can be done for a single file, or by concatenating several files for a single run. The output is either ASCII text or binary data, when saved into a file, ready to be processed again with the same tools.

You may have several netflow sources - let's say 'router1' 'router2' and so on. The data is organized as follows:

```
/flow_base_dir/router1
/flow_base_dir/router2
```

which means router1 and router2 are subdirs of the flow_base_dir.
For each of the netflow sources you have to start an nfcpad process:

```
nfcapd -w -D -l /flow_base_dir/router1 -p 23456
nfcapd -w -D -l /flow_base_dir/router2 -p 23457
```

A note on security: None of the tools requires root privileges, unless you have a port < 1024. However, there is no access control mechanism in nfcapd. It is assumed, that host level security is in place to filter the proper IP addresses.

See the manual pages or use the -h switch for details on using each of the programs.

1.3 Configuration:

You need to configure your router to export netflow data. See the relevant documentation for your model. A generic CISCO sample to enable Netflow on an interface may look like:

```
interface fastethernet 0/0
ip route-cache flow
```

To tell the router where to send the netflow data, enter the following global configuration command:

```
        ip flow-export
        ip flow-export version 5
        ip flow-cache timeout active 5
```

This breaks up long-lived flows into 5-minute segments. You can choose any number of minutes between 1 and 60, but should be equal or less than the file rotation period - typically 5 minutes.

On the 6500/7600, you must make sure to enable "NDE" in addition to normal Netflow export. NDE (Netflow Data Export) is the hardware variant of Netflow export on the Catalyst 6500/7600 OSR. Here's a configuration example:

```
        mls flow ip interface-full
        mls flow ipv6 interface-full
        mls nde sender version 5
```

Note that IPv6 NDE isn't implemented yet, but it can still be useful to be able to look at the "live" flows with "show mls netflow ipv6".

On a busy router, consider aggressively timing out small flows:

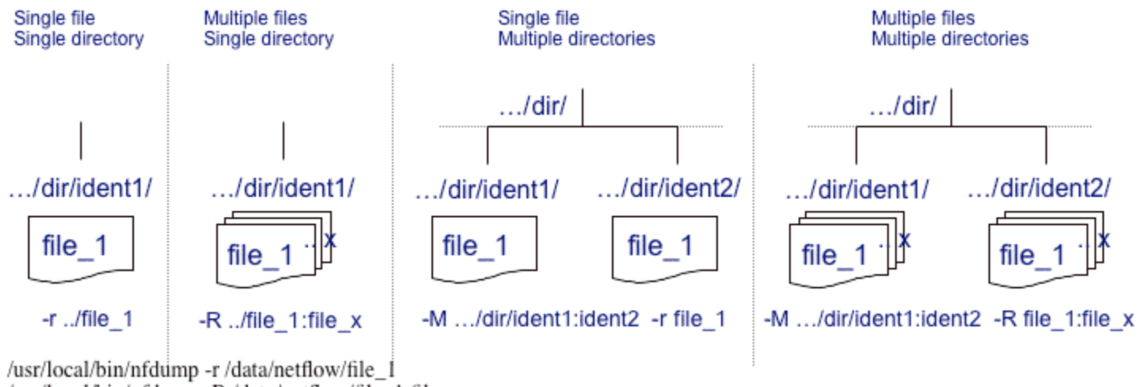
```
        mls aging fast time 4 threshold 2
        mls aging normal 32
        mls aging long 900
```

You still want the "traditional" Netflow configuration, including "ip flow ingress" or "ip route-cache flow" on every interface, so that you see "software-switched" flows such as those that go to the router itself. See the relevant documentation for a full description of netflow commands.

Note: Netflow version v5 and v7 have 32 bit counter values. The number of packets or bytes may overflow this value, within the flow-cache timeout on very busy routers. To prevent overflow, you may consider to reduce the flow-cache timeout to lower values. All nfdump tools use 64 bit counters internally, which means, all aggregated values are correctly reported.

1.4 Netflow Processing

Please have a look at the nfdump(1) man page for a detailed explanation of all options available.
Reading flow data:
Flows are read either from a single file or from a sequence of files:

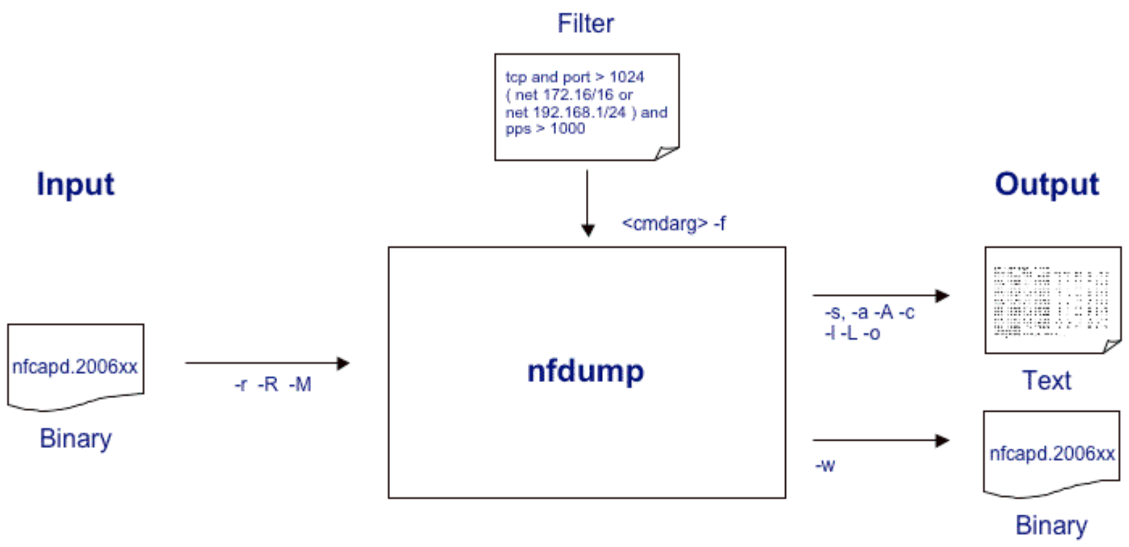


```
/usr/local/bin/nfdump -r /data/netflow/file_1
/usr/local/bin/nfdump -R /data/netflow/file_1:file_x
/usr/local/bin/nfdump -M /data/netflow/ident1:ident2 -r file_1
/usr/local/bin/nfdump -M /data/netflow/ident1:ident2 -R file_1:file_x
```

-r <single file>	Read a single file.
-R </path/to/first-file:lastfile>	Read data from a sequence of files from /path/to/first-file to /path/to/last-file
-M /path/to/first-dir:next-dir:last-dir -r <single-file> -M /path/to/first-dir:next-dir:last-dir -R <first-file:last-file>	Read from a sequence of files from several directories: File sequence is: /path/to/first-dir/single-file /path/to/next-dir/single-file /path/to/last-dir/single-file or /path/to/first-dir/first-file .... /path/to/first-dir/last-file /path/to/next-dir/first-file .... /path/to/next-dir/last-file /path/to/last-dir/first-file .... /path/to/last-dir/last-file

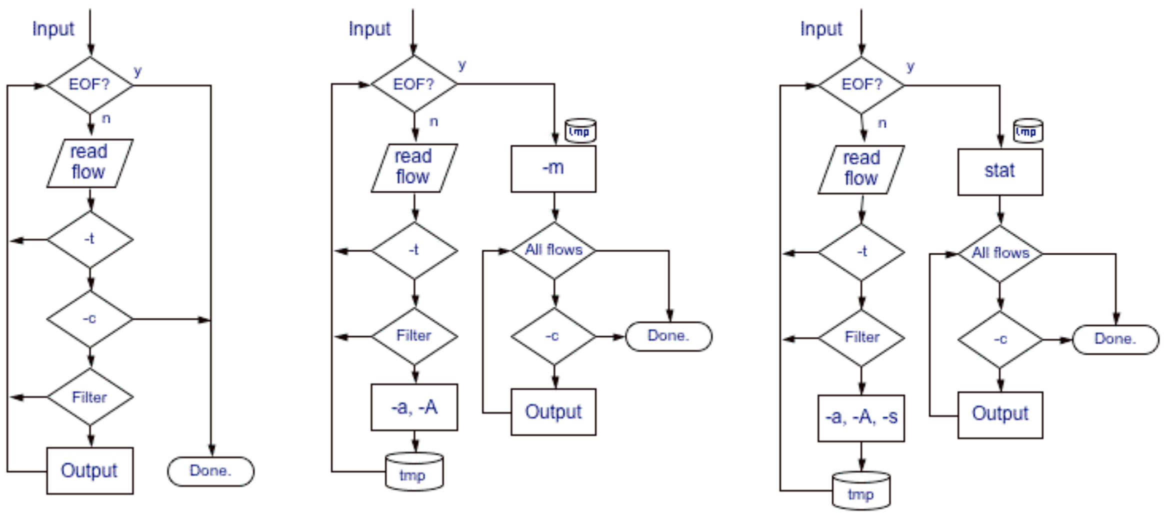
Specials:

-R </path/to/directory> -M /path/to/first-dir:next-dir:last-dir -R .	Read all files in a directory
-R </path/to/first-file> -M /path/to/first-dir:next-dir:last-dir -R <first-file>	Read all files in a directory starting with a given file



Writing flow data:
Processed flows can be either printed in ASCII to stdout or written to a file. The binary file can be read again by nfdump for further processing.

The diagrams below show at which point various options affect the netflow processing for normal flow listings, date sorted listings and statistic outputs.



1.5 Output formats:

nfdump has four fixed output formats: ***raw***, ***line***, ***long*** and ***extended***. However the user may specify any desired output format using the custom output format ***fmt****:...*
The default format is *line*, unless otherwise specified.

Raw format:
The raw format displays each record in multiple lines, and prints any available information in the record. The record printed is netflow version independent, but may contain different additional fields depending on the source:

```
Flow Record:
  Flags        =       0x00000000
  size         =               52
  mark         =                0
  srcaddr      =    36.249.80.226
  dstaddr      =    92.98.219.116
  First        =       1125377992 [2005-08-30 06:59:52]
  Last         =       1125377992 [2005-08-30 06:59:52]
  msec_first   =              338
  msec_last    =              338
  dir          =                0
  tcp_flags    =                0
  prot         =               17
  tos          =                0
  input        =                5
  output       =                3
  srcas        =             1299
  dstas        =                0
  srcport      =             3040
  dstport      =             1434
  dPkts        =                1
  dOctets      =              404
```

This format is rarely used, but contains any information available for this record.

Line Format: -o line
This is the default format and shows one netflow record per line:

```
Date flow start         Duration Proto  Src IP Addr:Port         Dst IP Addr:Port   Packets    Bytes Flows
2005-08-30 06:59:52.338    0.001 UDP  36.249.80.226:3040  ->   92.98.219.116:1434         1      404     1
```

The date and duration of the flow are given in millisecond resolution. The number of flows is always 1 unless flows are aggregated. See below.

Long format: -o long
This format contains additional information such as TCP flags, Type of Service (Tos) etc:

```
Date flow start     Duration Proto    Src IP Addr:Port      Dst IP Addr:Port Flags Tos Packets Bytes Flows
2005-08-30 06:53:53.370 63.545 TCP 113.138.32.152:25    -> 222.33.70.124:3575 .AP.SF   0     62  3512     1
2005-08-30 06:53:53.370 63.545 TCP  222.33.70.124:3575 -> 113.138.32.152:25   .AP.SF   0     58  3300     1
```

Extended format: **-o extended**
This format contains additional information to format long: pps (packets per second) bps (bits per second) and bps (bytes per packet) are calculated and displayed for each record. For displaying purpose, the start time has been suppressed to fit the flow on one line:

```
… Duration Proto  Src IP Addr:Port       Dst IP Addr:Port  Flags Tos Packets Bytes pps bps Bpp Flows
…   63.545 TCP 113.138.32.152:25   ->  222.33.70.124:3575 .AP.SF 0        62  3512   0 442  56     1
…   63.545 TCP  222.33.70.124:3575 ->  113.138.32.152:25  .AP.SF 0        58  3300   0 415  56     1
```

Custom output format: **-o fmt:**..
This is the most flexible format, as you can specify yourself how the output looks like. The output format is defined using element tags as well as plain ASCII text.

Predefined element tags:

Tag	**Description**	**Tag**	**Description**
%ts	Start Time - first seen	%in	Input Interface num
%te	End Time - last seen	%out	Output Interface num
%td	Duration	%pkt	Packets
%pr	Protocol	%byt	Bytes
%sa	Source Address	%fl	Flows
%da	Destination Address	%pkt	Packets
%sap	Source Address:Port	%flg	TCP Flags
%dap	Destination Address:Port	%tos	Tos
%sp	Source Port	%bps	bps - bits per second
%dp	Destination Port	%pps	pps - packets per second
%sas	Source AS	%bpp	bps - Bytes per package
%das	Destination AS		

Example: The format -o long can be described as:

```
        -o “fmt:%ts %td %pr %sap -> %dap %pkt %byt %fl"
```

Often used output formats can be compiled into nfdump for easy access. See *nfdump.c* source file for defining more output formats.

Printing IPv6 records:
IPv6 addresses need much more space to display than IPv4 addresses. In order to keep the output clearly arranged, IPv6 addresses are shrunk in normal output.

```
Date flow start       Duration Proto Src IP Addr:Port        Dst IP Addr:Port Packets Bytes Flows
2006-03-09 11:55:03.900  0.000 ICMP6 2005:62..2c:9c10.0 -> 2005:62..c000::d.0       1   104     1
```

The middle part of each IPv6 address is cut, but should allow to identify addresses, though. If the full length of IPv6 addresses is required, add the digit '6' to the output format (e.g. **-o line6, -o long6, -o extended6**) or add the option -6 (e.g. **-o extended -6**).

```
… Duration Proto                      Src IP Addr:Port        Dst IP Addr:Port Packets    Bytes Flows
… 0.000    ICMP6  2005:620:0:8:203:baff:fe2c:9c10.0 -> 2005:620:0:c000::d.0          1      104     1
```

Aggregating Flows: -a [-A <scheme>]
Flows can be aggregated by specifying -a. By default, flows with identical protocol and identical source and destination IP address as well as identical source and destination ports are aggregated. However, this behaviour can be changed by specifying a different aggregation scheme with -A. The option -A accepts any combination *srcip, dstip, srcport, dstport*.

Examples:

Default aggregation: 10 flows aggregated.

```
Date flow start        Duration Proto   Src IP Addr:Port       Dst IP Addr:Port  Packets  Bytes Flows
2005-08-30 06:59:54.324 250.498 TCP   63.183.112.97:9050  -> 146.69.72.180:51899      12   2198    10
```

Aggregate source IP address and destination port: **-A srcip,dstport**

```
Date flow start       Duration Proto  Src IP Addr:Port Dst IP Addr:Port Flags Tos Packets  Bytes   pps    bps  Bpp Flows
2005-08-30 06:59:25.137 213.697 TCP    32.249.32.48:0 -> 0.0.0.0:135   ......   0       23   1104     0     41   48    13
2005-08-30 06:59:24.563 330.110 TCP  49.112.228.156:0 -> 0.0.0.0:1433  ......   0    47943  2.2 M   145  55769   48 27864
2005-08-30 06:59:54.322 201.857 TCP 148.190.164.126:0 -> 0.0.0.0:36129 ......   0       10    460     0     18   46     6
2005-08-30 06:59:54.257  48.768 TCP     92.90.57.46:0 -> 0.0.0.0:59501 ......   0        5    230     0     37   46     2
```

All other elements, not aggregated are set to '0'.

Subnet aggregation.
It is also possible to aggregate flows on a subnet level. In order to create appropriate masks for aggregation, the protocol version is required with the address field:
Example: **-a -A srcip4/24, dstport** aggregates flows on a /24 IPv4 base and destination port.

1.6 Filter Syntax

nfdump has a powerful and fast filter engine. All flows are filtered before they are further processed. If no filter is given, any flow will be processed. The filter is either given on the command line as last argument enclosed in **'**, or in a file. Any line in the file starting with a **#** is treated as a comment. The filter syntax is similar to the tcpdump syntax.

Any filter consists of one or more expressions **expr**. Any number of expr can be linked together:

expr and expr, **expr or expr**, **not expr**, **(expr)**.

expr can be one of the following filter primitives:

protocol version
: **inet** or **ipv4** for IPv4 and **inet6** or **ipv6** for IPv6 flows only

protocol
: **TCP, UDP, ICMP, GRE, ESP, AH, RSVP** etc. or **PROTO <num>** where num is the protocol number.

IP address
: [SourceDestination] **IP a.b.c.d or**
[SourceDestination] **HOST a.b.c.d** with a.b.c.d as any valid IP address. SourceDestination may be omitted.

SourceDestination
: defines the IP address to be selected and can be **SRC, DST** or any combination of **SRC and|or DST**. Omitting SourceDestination is equivalent to **SRC or DST**.

network
: [SourceDestination] **NET a.b.c.d m.n.r.s**
[SourceDestination] **NET a.b.c.d / num** with a.b.c.d as network number, m.n.r.s as netmask or num as maskbits respectively.   The network may be given as **a.b**, **a.b.c**, where a B or C-class equivalent netmask is assumed.

Port
: [SourceDestination] **PORT** [comp] **num** with num as a valid port number. If comp is omitted, '=' is assumed.

Interface
: [inout] **IF num** with num as an interface number.

inout
: defines the interface to be selected and can be **IN** or **OUT**.

Flags
: **flags tcpflags** with tcpflags as a combination of:

A ACK.
S SYN.
F FIN.
R Reset.
P Push.
U Urgent.
X All flags on.
The ordering of the flags is not relevant. Flags not mentioned are treated as don't care. In order to get those flows with only the SYN flag set, use the syntax '**flags S and not flags AFRPU**'.

TOS Type of service: **tos value** with value 0..255.

Packets
packets [comp] **num** [scale] to specify the packet count in the netflow record.

Bytes
bytes [comp] **num** [scale] to specify the byte count in the netflow record.

Packets per second: Calculated value.
pps [comp] **num** [scale] to specify the pps of the flow.

Duration: Calculated value
duration [comp] **num** to specify the duration in milliseconds of the flow.

Bits per second: Calculated value.
bps [comp] **num** [scale] to specify the bps of the flow.

Bytes per packet: Calculated value.
bpp [comp] **num** [scale] to specify the bpp of the flow.

AS [SourceDestination] **AS num** with num as a valid AS number.

scale Scaling factor. Maybe **k m g**. Factor is 1024

comp The following comparators are supported:
=, ==, >, <, EQ, LT, GT . If comp is omitted, **'='** is assumed.

Examples:

```
nfdump -r /any/dir/nfcapd.200407110845 -c 100 'tcp and \
( src ip 172.16.17.18 or dst ip 172.16.17.19 )'

nfdump -r /and/dir/nfcapd.200407110845 -A srcip,dstport 'in if 5 and \
net 10.0.0.0/24 and not host 10.0.0.1 and bps > 10k and duration < 100 and dst port 1433'
```

Top N Statistics: [-n <num>] -s type[/orderby]
nfdump provides a number of statistics. These can be requested be supplying one or more -s arguments: **-s type[/orderby]** where as type can be:

record	Statistic about aggregated netflow records.
srcip	Statistic about source IP addresses
dstip	Statistic about destination IP addresses
ip	Statistic about any (source or destination) IP addresses
srcport	Statistic about source ports
dstport	Statistic about destination ports
port	Statistic about any (source or destination) ports
srcas	Statistic about source AS numbers
dstas	Statistic about destination AS numbers
as	Statistic about any (source or destination) AS numbers
inif	Statistic about input interface numbers
outif	Statistic about output interface numbers

if	Statistic about any (input or output) interface numbers
proto	Statistic about protocol numbers

orderby is optional and specifies the order by which the statistics is ordered and can be **flows, packets, bytes, pps, bps** or **bpp**. You may specify more than one orderby which results in the same statistic but ordered differently. If no orderby is given, statistics are ordered by flows. You can specify as many -s arguments on the command line for the same run.

The record statistics can be formatted according to the available output formats given by -o (see above). Top N defaults to 10 unless specified otherwise by supplying **-n <num>. -n 0** means unlimited number, unless for **-s record**, which n is limited to 1000.

Example:

```
nfdump -r nfcapd.200508300700 -o extended -s srcip -s ip/flows -s dstport/pps/packets/bytes -s record/bytes
Aggregated flows 850332
Top 10 flows ordered by bytes:
Date flow start          Duration Proto  Src IP Addr:Port        Dst IP Addr:Port  Flags Tos Packets Bytes  pps   bps    Bpp Fl
2005-08-30 06:50:11.218  700.352 TCP   126.52.54.27:47303 ->   42.90.25.218:435   ...... 0  1.4 M   2.0 G 2023 5.6 M   1498 1
2005-08-30 06:47:06.504  904.128 TCP 198.100.18.123:54945 ->   126.52.57.13:119   ...... 0 567732 795.1 M  627 2.5 M   1468 1
2005-08-30 06:47:06.310  904.384 TCP   126.52.57.13:45633 -> 91.127.227.206:119   ...... 0 321148 456.5 M  355 4.0 M   1490 1
2005-08-30 06:47:14.315  904.448 TCP   126.52.57.13:45598 -> 91.127.227.206:119   ...... 0 320710 455.9 M  354 4.0 M   1490 1
2005-08-30 06:47:14.316  904.448 TCP   126.52.57.13:45629 -> 91.127.227.206:119   ...... 0 317764 451.5 M  351 4.0 M   1489 1
2005-08-30 06:47:14.315  904.448 TCP   126.52.57.13:45634 -> 91.127.227.206:119   ...... 0 317611 451.2 M  351 4.0 M   1489 1
2005-08-30 06:47:06.313  904.384 TCP   126.52.57.13:45675 -> 91.127.227.206:119   ...... 0 317319 451.0 M  350 4.0 M   1490 1
2005-08-30 06:47:06.313  904.384 TCP   126.52.57.13:45619 -> 91.127.227.206:119   ...... 0 314199 446.5 M  347 3.9 M   1490 1
2005-08-30 06:47:06.321  790.976 TCP   126.52.54.35:59898 ->  132.94.115.59:246   ...... 0 254717 362.4 M  322 3.7 M   1491 1
2005-08-30 06:47:14.316  904.384 TCP   126.52.54.35:59773 -> 55.107.224.187:11709 ...... 0 272710 348.5 M  301 3.1 M   1340 1

Top 10 Src IP Addr ordered by flows:
Date first seen          Duration      Src IP Addr    Flows  Packets     Bytes      pps      bps    bpp
2005-08-30 06:45:50.990 1147.332  125.67.123.234   109183   202523    13.1 M      176    96116     68
2005-08-30 06:45:02.928 1192.834  94.180.151.203    62794   219715    25.9 M      184   182294    123
2005-08-30 06:59:24.563  330.110     9.209.28.173    27864    47943     2.2 M      145    55769     48
2005-08-30 06:45:07.728 1190.594  125.248.33.146    17271    41942     5.7 M       35    40438    143
2005-08-30 06:59:16.431  341.892    138.5.122.251    12253    75925    39.2 M      222   962768    541
2005-08-30 06:59:48.111  310.211   130.195.23.210    11742    46928     3.2 M      151    86940     71
2005-08-30 06:59:54.066  304.257    255.93.216.43    11383    56943     4.5 M      187   123968     82
2005-08-30 06:59:53.362  304.894    219.182.16.57    11209    44784     2.0 M      146    54640     46
2005-08-30 06:47:06.503 1068.361       3.15.99.52     9000    16962     3.1 M       15    24415    192
2005-08-30 06:59:52.784  172.102  11.121.123.165     7176     7176    330096       41    15344     46

Top 10     IP Addr ordered by flows:
Date first seen          Duration          IP Addr    Flows  Packets     Bytes      pps      bps    bpp
2005-08-30 06:45:50.990 1147.332  125.67.123.234   234366   458197    30.2 M      399   221164     69
2005-08-30 06:45:02.928 1192.835  94.180.151.203   115841   428885    42.6 M      359   299577    104
2005-08-30 06:45:07.728 1190.594  125.248.33.146    28218    73178     7.8 M       61    55234    112
2005-08-30 06:59:24.563  330.110     9.209.28.173    27916    48086     2.2 M      145    55931     47
2005-08-30 06:59:48.111  310.212   130.195.23.210    23467   105779     7.9 M      340   212311     77
2005-08-30 06:59:53.362  304.960    219.182.16.57    22938    89563     4.0 M      293   109251     46
2005-08-30 06:59:54.066  304.257    255.93.216.43    22769   102496     7.6 M      336   210205     77
2005-08-30 06:59:16.431  341.892    138.5.122.251    21840   132119    45.3 M      386    1.1 M    359
2005-08-30 06:46:33.104 1102.656  129.251.42.241    20305    74505     5.8 M       67    43757     80
2005-08-30 06:47:07.272 1068.360 131.250.225.247    14452    36714     3.1 M       34    24078     87

Top 10    Dst Port ordered by packets:
Date first seen          Duration        Dst Port    Flows  Packets     Bytes      pps      bps    bpp
2005-08-30 06:45:55.150 1129.287             119       99    3.3 M     4.7 G     3079   34.1 M   1450
2005-08-30 06:45:47.858 1150.465              80    56282    1.4 M   135.1 M     1307   984959     94
2005-08-30 06:47:06.375 1032.270             435        5    1.4 M     2.0 G     1410   16.0 M   1488
2005-08-30 06:45:41.715 1157.052               0    40088    1.4 M   261.0 M     1225    1.8 M    192
2005-08-30 06:45:26.415 1171.905            6881     8898   592649   545.1 M      505    3.7 M    964
2005-08-30 06:47:06.310 1032.335             433       13   588268   814.3 M      569    6.3 M   1451
2005-08-30 06:45:02.928 1195.523              53   140178   481356    37.1 M      402   260537     80
2005-08-30 06:44:59.090 1199.038            4662     9238   344122   267.1 M      286    1.8 M    813
2005-08-30 06:45:50.990 1144.773             123   176044   302564    23.0 M      264   168866     79
2005-08-30 06:47:14.316  939.333           11709        4   272713   348.5 M      290    3.0 M   1339

Top 10    Dst Port ordered by bytes:
Date first seen          Duration        Dst Port    Flows  Packets     Bytes      pps      bps    bpp
2005-08-30 06:45:55.150 1129.287             119       99    3.3 M     4.7 G     3079   34.1 M   1450
2005-08-30 06:47:06.375 1032.270             435        5    1.4 M     2.0 G     1410   16.0 M   1488
2005-08-30 06:47:06.310 1032.335             433       13   588268   814.3 M      569    6.3 M   1451
2005-08-30 06:45:26.415 1171.905            6881     8898   592649   545.1 M      505    3.7 M    964
2005-08-30 06:47:06.321 1053.251            2466       30   255460   363.4 M      242    2.8 M   1491
2005-08-30 06:47:14.316  939.333           11709        4   272713   348.5 M      290    3.0 M   1339
2005-08-30 06:48:42.325  904.448           52911        5   191559   274.0 M      211    2.4 M   1499
2005-08-30 06:44:59.090 1199.038            4662     9238   344122   267.1 M      286    1.8 M    813
2005-08-30 06:45:41.715 1157.052               0    40088    1.4 M   261.0 M     1225    1.8 M    192
2005-08-30 06:47:06.313 1065.928            1101       83   173933   248.4 M      163    1.9 M   1497

Top 10    Dst Port ordered by pps:
Date first seen          Duration        Dst Port    Flows  Packets     Bytes      pps      bps    bpp
2005-08-30 06:45:55.150 1129.287             119       99    3.3 M     4.7 G     3079   34.1 M   1450
2005-08-30 07:02:55.248    0.002           39601        3        5       748     2499    2.9 M    149
2005-08-30 07:00:28.817    0.512           54286        1     1279     66882     2498  1045031     52
2005-08-30 06:47:06.375 1032.270             435        5    1.4 M     2.0 G     1410   16.0 M   1488
2005-08-30 06:45:47.858 1150.465              80    56282    1.4 M   135.1 M     1307   984959     94
2005-08-30 06:45:41.715 1157.052               0    40088    1.4 M   261.0 M     1225    1.8 M    192
2005-08-30 07:00:28.305    0.002           56997        2        2        92      999   367999     46
```

```
2005-08-30 07:03:55.859    0.064           47264        1        40    49456       624    5.9 M  1236
2005-08-30 06:47:06.310 1032.335             433       13    588268  814.3 M       569    6.3 M  1451
2005-08-30 07:02:44.692    0.064            8612        1        33    41848       515    5.0 M  1268

Time window: Aug 30 2005 06:44:54 - Aug 30 2005 07:04:58
Flows analysed: 1115890 matched: 1115890, Bytes read: 54486168
Sys: 2.286s flows/second: 488001.9   Wall: 2.386s flows/second: 467490.9
```

1.7 Other Options

Anonymizing Flows: -K <key>
IP addresses in flows can be anonymised by supplying -K <key>. nfdump uses the Crypto-PAn module to anonymise IP addresses. See http://www.cc.gatech.edu/computing/Telecomm/cryptopan/ for further details of Crypto-PAn. **key** is either a 32 character string or a 64 digit hex string starting with 0x. IP addresses are anonymised before they are printed or saved to file. This means the filter applies to the original IP address.

Converting flow-tools netflow data:
The flow-tools converter reads flow-tools data either from stdin, or from a given file (-r). It converts the data into nfdump format and writes nfdump records to stdout.
To concert a file: ft2nfdump -r <flow-tools-file> | nfdump -w <nfdump-file>. Of course you can supply any other nfdump command line switches to directly process flow-tools data with nfdump.

2 NfSen - Netflow Sensor

NfSen is a graphical web based front end for the nfdump netflow tools.
NfSen allows you to:

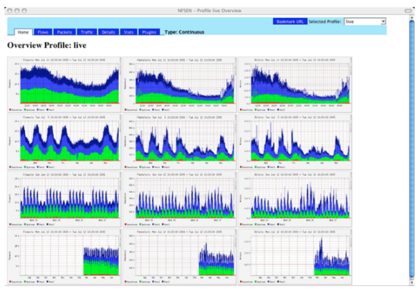
- Display your netflow data: Flows, Packets and Bytes using RRD (Round Robin Database).
- Easily navigate through the netflow data.
- Process the netflow data within the specified time span.
- Create history as well as continuous profiles.
- Write your own plugins to process netflow data on a regular interval.

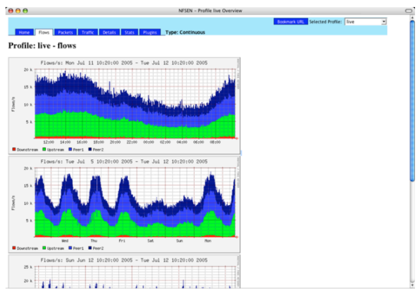
Different tasks need different interfaces to your netflow data. NfSen allows you to keep all the convenient advantages of the command line using nfdump directly and gives you also a graphical overview over your netflow data.

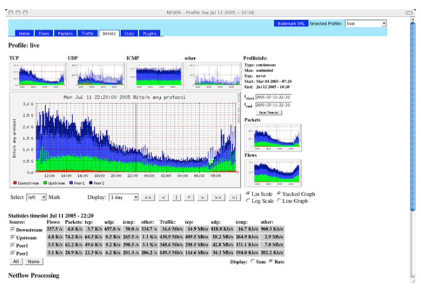
Note: All IP addresses in this document are anonymised.

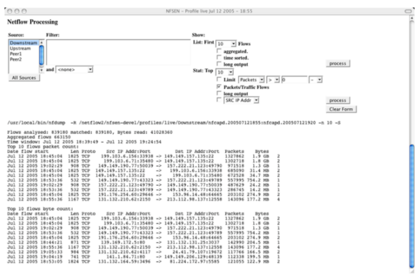
This documentation describes v1.2.3 of NfSen.

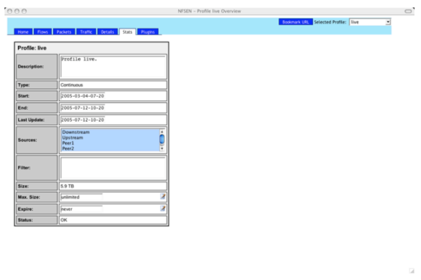
2.1 Screen Shots


NfSen - General Overview Page


NfSen - Flow Overview Page


NfSen - Navigation Page


NfSen - Netflow Processing output


NfSen - Profile Info

2.2 Installing NfSen

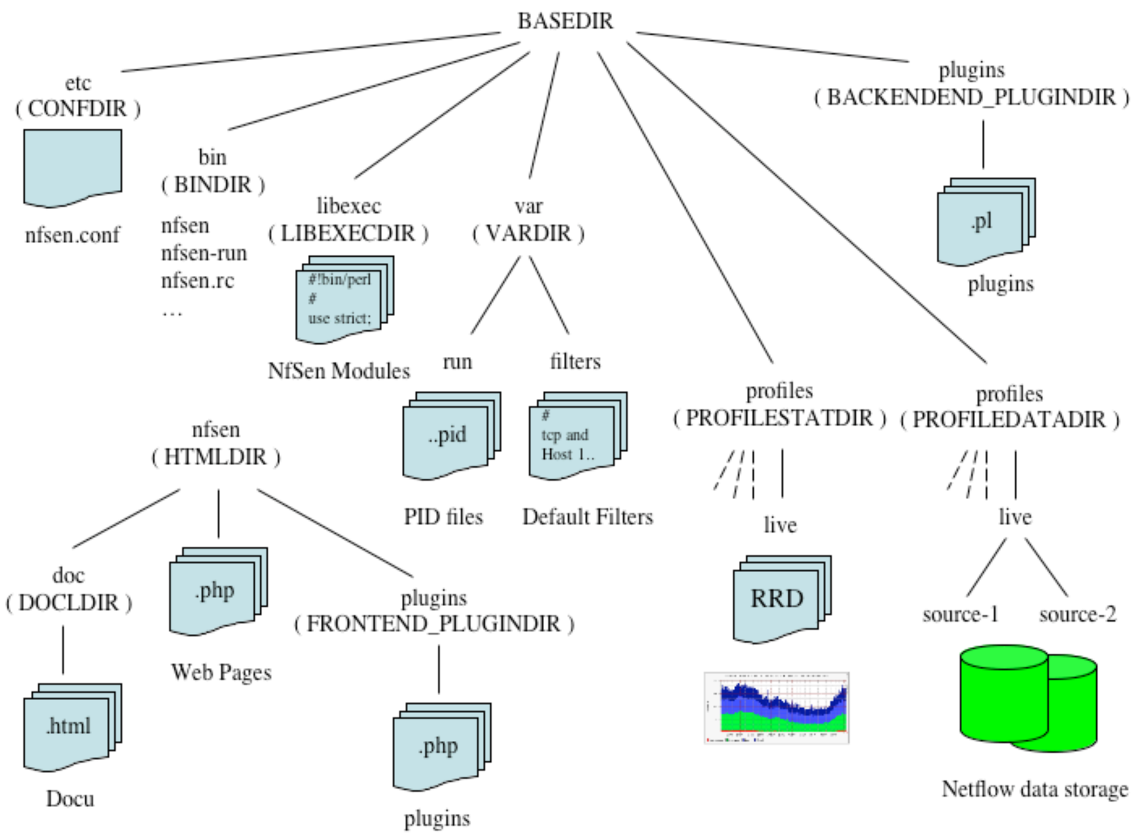
2.2.1 Prerequisites:

- PHP and Perl:
  NfSen is written in PHP and Perl and should run on any *NIX system. At least Perl 5.6.0 and PHP > 4.1 is required including the Perl regex extension.
- RRD tools
  For the netflow graphs, NfSen requires the RRD tools, at least the RRDs Perl Module.

- Nfdump tools
  The nfump tools are the backend tools for NfSen and will collect and process the netflow data. Make sure you have at least version 1.4. You can download nfdump from sourceforge nfdump.sourceforge.net.

2.2.2 First Installation

NfSen has a very flexible directory layout. To simplify matters, the default layout stores everything but the html pages under BASEDIR. However, you may configure NfSen to fit your local needs. The figure below shows the default layout with all configurable directories.
All netflow data is stored under PROFILEDATADIR. So make sure you have enough disk space for this directory.



Directory Structure

If you have installed all prerequisites, change to the etc directory and copy the NfSen template config file nfsen-dist.conf to nfsen.conf.

Edit nfsen.conf according your needs and setup:

Master Config File:

```
##############################
#
# NfSen master config file
#
# Configuration of NfSen:
# Set all the values to fit your NfSen setup and run the 'install.pl'
# script from the nfsen distribution directory.
#
# You should not need to changes anything after NfSen is installed,
# besides the NfSen plugins at the bottom.
# When you make any changes in the plugins section, run 'nfsen reload'
# to make sure nfsen-run gets notified about your plugins.
#
# Do not change any other settings after NfSen is installed.
# otherwise you must rerun the install.pl script.
#
# The syntax must conform to Perl syntax.
#
##############################
#
# NfSen default layout:
# Any scripts, modules or profiles are installed by default under $BASEDIR.
# However, you may change any of these settings to fit your requested layout.

#
# Required for default layout
$BASEDIR = "/data/nfsen";

#
# Where to install the NfSen binaries
$BINDIR="${BASEDIR}/bin";

#
# Where to install the NfSen Perl modules
$LIBEXECDIR="${BASEDIR}/libexec";

#
# Where to install the config files
$CONFDIR="${BASEDIR}/etc";

#
# NfSen html pages directory:
# All php scripts will be installed here.
# URL: Entry point for nfsen: http://<webserver>/nfsen/nfsen.php
$HTMLDIR = "/var/www/nfsen/";

#
# Where to install the docs
$DOCDIR="${HTMLDIR}/doc";

#
# Var space for NfSen
$VARDIR="${BASEDIR}/var";

#
# The Profiles stat directory, where all profile information
# RRD DBs and gif pictures of the profile are stored
$PROFILESTATDIR="${BASEDIR}/profiles";

#
# The Profiles directory, where all netflow data is stored
$PROFILEDATADIR="${BASEDIR}/profiles";

#
# Where go all the backend plugins
$BACKEND_PLUGINDIR="${BASEDIR}/plugins";

#
# Where go all the frontend plugins
$FRONTEND_PLUGINDIR="${HTMLDIR}/plugins";

#
# nfdump tools path
$PREFIX = '/usr/local/bin';

#
# BASEDIR unrelated vars:
#
```

```
# Run nfcapd as this user
# This may be a different or the same uid than your web server.
# Note: This user must be in group $WWWGROUP, otherwise nfcapd
# is not able to write data files!
$USER = "netflow";

# user and group of the web server process
# All netflow processing will be done with this user
$WWWUSER = "www";
$WWWGROUP = "www";

# Receive buffer size for nfcapd - see man page nfcapd(1)
$BUFFLEN = 200000;

# Netflow sources
# Define an ident string, port and colour per netflow source
# ident identifies this netflow source. e.g. the router name,
# Upstream provider name etc.
# port nfcapd listens on this port for netflow data for this source
# col colour in nfsen graphs for this source
#
# Syntax:
# 'ident' => { 'port' => '<portnum>', 'col' => '<colour>' }
# Ident strings must be 1 to 19 characters long only, containing characters [a-zA-Z0-9_].

%sources = (
         'upstream1' => { 'port' => '9995', 'col' => '#0000ff' },
         'peer1' => { 'port' => '9996', 'col' => '#ff0000' },
);

#
# Low water mark: When expiring files, delete files until
# size = max size * low water mark
# typically 0.9
$low_water = 0.9;

#
# syslog facility for periodic jobs
# nfsen uses level 'debug', 'info', 'warning' and 'err'
# Note: nfsen is very chatty for level 'debug' and 'info'
# For normal operation, you may set the logging level in syslog.conf
# to warning or error unless you want to debug NfSen
$syslog_facility = 'local3';

#
# plugins
# plugins are run for each timeslot, after the roll over of new data files.
# A plugin may run for any profile or for a specific profile only.
# Syntax: [ 'profile', 'module' ]
# profile: ',' separated list of profiles, or "*' for any profile
# module: Module name.
# The module follows the standard Perl module conventions, with at least two
# additional functions: Init() and run(). See demoplugin.pm for a simple template.
# Plugins are installed under
#
# $BACKEND_PLUGINDIR and $FRONTEND_PLUGINDIR

@plugins = (
 # profile # module
 # [ '*', 'demoplugin' ],
);

#
# Notification module
# The Notification module is an optional module. If you want your plugins to
# notify any result by email, use this module.
# Make sure you have installed Mail::Internet before using the module
#
# Use this from address
$MAIL_FROM = 'your@from.example.net';

# Use these recipients
$RCPT_TO = 'any@example.net, another@example.net';

# Use this SMTP server
$SMTP_SERVER = 'localhost';

1;
```

When you are done with nfsen.conf, run the install.pl script in the NfSen distribution directory:

```
./install.pl etc/nfsen.conf
```

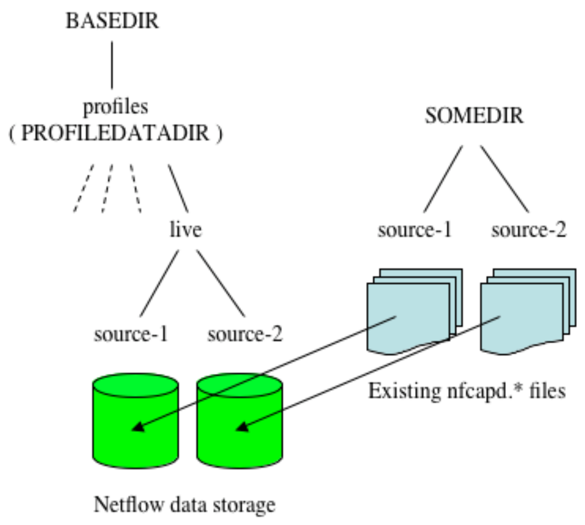
Running install.pl will:

- Create the NfSen environment under BASEDIR
- Copy the php/html files into the HTMLDIR
- Create the live profile.
- Prepares the RRD DBs for the live profile.
- Creates and configures config.php

After the installation, you will find the nfsen.conf file in CONFDIR. The documentation, is installed in DOCDIR. If you want the document available as Help link in the Web Frontend, uncomment line 18 in navigation.php:

```
// print "<a href='doc/NfSen.html' target='_blank' >Help</a>\n";
```

2.2.3 Importing existing netflow data



If you have existing netflow data from nfdump, follow these steps:

- cd BASEDIR/profiles/live
- Copy already existing data into the appropriate directory or make sure you have a soft link from the source directory to your flows.
- cd BASEDIR/bin and rebuild the profile live:
  ./nfsen -r live

The live profile is now setup with your existing data. You can verify your profile: ./nfsen -l live. The status of the profile is set to 'rebuild' which will change automatically to 'OK' the first time the periodic task of nfsen-run is executed.

Make sure to disable your current start/stop script, as NfSen will provide it's own start/stop script to start all required nfcapd processes, as well as the nfsen-run background process. The nfdump cleanup script nfclean.pl isn't needed either, as expiring the netflow data is fully integrated into NfSen.

2.2.4 Start-Stop NfSen

NfSen provides a start-stop script **nfsen.rc** in BINDIR. You may create a soft link from your appropriate rc.d directory to this file.

To start NfSen:

```
        BINDIR/nfsen.rc start
```

This starts all nfcapd processes to collect the netflow data and the nfsen-run background process to update your profiles, as new data becomes available. Point your web browser to nfsen.php. (Typically http://yourserver/nfsen/nfsen.php).

The background task nfsen-run as well as nfcapd log to syslog. nfsen-run is very chatty, when configuring syslog priority **'info'** or less. You may want to set the syslog priority to **'warning'** for normal operation. For debugging purpose, use **'info'** or **'debug'**.

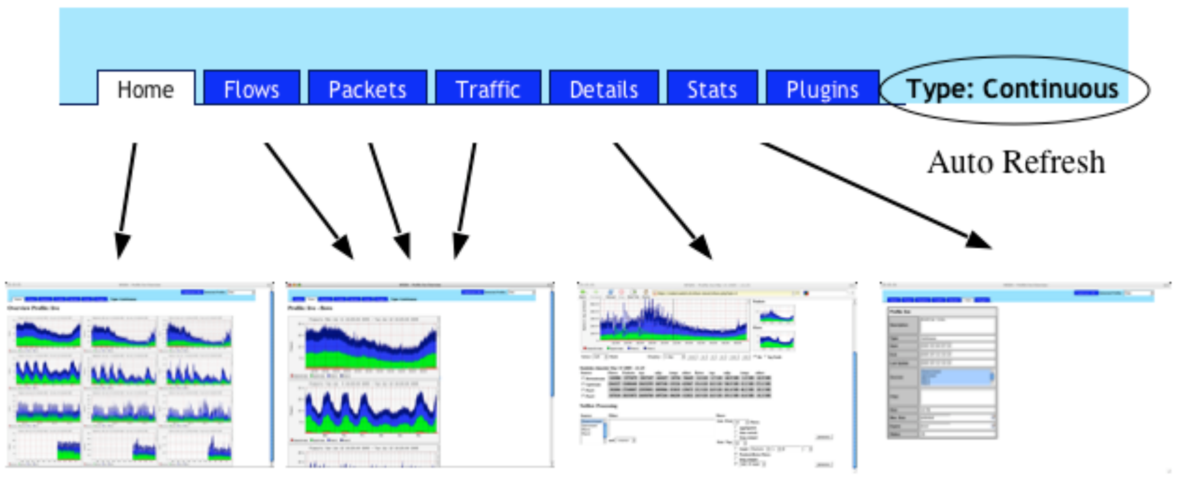
2.3 Working with NfSen

NfSen has two different user interfaces:

- Web Interface
- Command line interface

Most of the time you will want to use the web interface. However, you can do everything from the command line as well.

2.3.1 Views

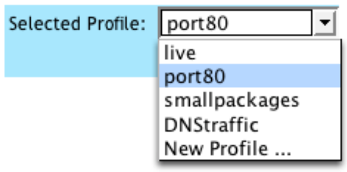


Tab Navigation

NfSen offers different views. Each of the views can be selected using the tabs at the top of the page. When you point your browser to the nfsen.php page, The 'Home' view is the default view and shows an overview of the currently selected profile. The three columns show the 'Flows', 'Packets' and 'Bytes' history. To select a different view, click into any graph or select the view in the tab, e.g. clicking into the column of the bytes history switches to the bytes view. Clicking into any graph in the 'Flows', 'Packets' or 'Bytes' view switches to the 'Detailed' view for a further analysis of the netflow data. If the currently selected profile is a continuous profile, the history pages are automatically refreshed every 5 minutes to update the graphs. This allows you to have a browser window on your screen, with always up to date graphs. Detailed information about the currently selected profile is available under the 'Stats' tab.

2.3.2 Profiles

A profile is a specific view on the netflow data. A profile is defined by its **name**, **type** and profile **filter**, which is any valid filter accepted by nfdump. At least the profile '**live**' is always available and is used to store your incoming netflow data without filtering. You can switch back and forth to any profile using the pull down menu in the upper right corner of the web page.

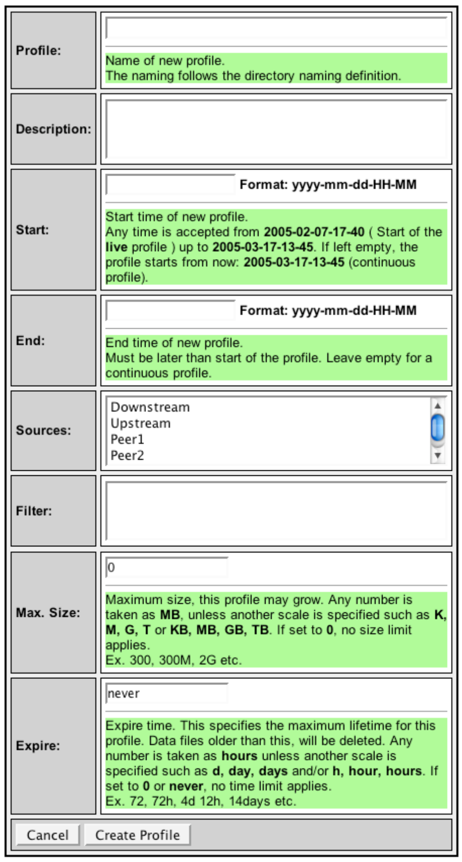


Profile Selection

2.3.3 Profile Types

A profile can be either of type **History** or **Continuous**. A history profile starts and ends back in the past and remains static. It neither grows nor expires. A continuous profile may start in the past and is continually updated while new netflow data becomes available. It grows dynamically and may have its own expire values set. Old data expires after a given amount of time or when the profile reaches a certain size.

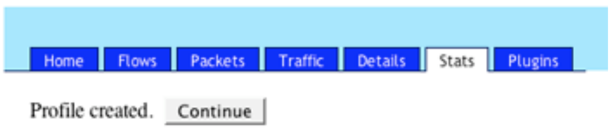
2.3.4 Creating profiles



Profile:	Name of new profile. The naming follows the directory naming definition.
Description:	
Start:	**Format: yyyy-mm-dd-HH-MM** Start time of new profile. Any time is accepted from **2005-02-07-17-40** (Start of the **live** profile) up to **2005-03-17-13-45**. If left empty, the profile starts from now: **2005-03-17-13-45** (continuous profile).
End:	**Format: yyyy-mm-dd-HH-MM** End time of new profile. Must be later than start of the profile. Leave empty for a continuous profile.
Sources:	Downstream Upstream Peer1 Peer2
Filter:	
Max. Size:	0 Maximum size, this profile may grow. Any number is taken as **MB**, unless another scale is specified such as **K, M, G, T** or **KB, MB, GB, TB**. If set to **0**, no size limit applies. Ex. 300, 300M, 2G etc.
Expire:	never Expire time. This specifies the maximum lifetime for this profile. Data files older than this, will be deleted. Any number is taken as **hours** unless another scale is specified such as **d, day, days** and/or **h, hour, hours**. If set to **0** or **never**, no time limit applies. Ex. 72, 72h, 4d 12h, 14days etc.
Cancel	Create Profile

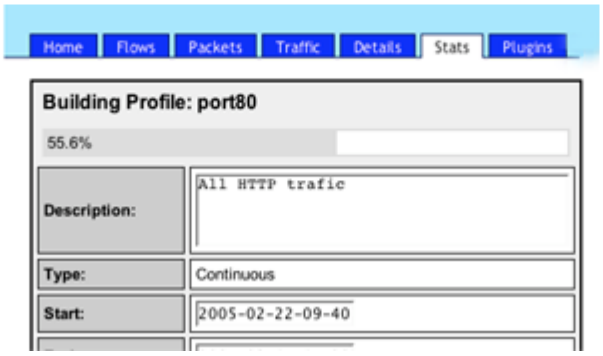
Select the "**New profile ...**" entry in the profile pull down menu.

Complete the 'New Profile' form to start building the profile. The profile type is automatically detected according the '**Start**' and '**End**' values you enter. The help text should guide you through the process of creating the profile. As profiles are created from netflow data from profile 'live', the start and end of the profile must fall in the time range of the profile 'live'.



Successful creation of new profile.

When the profile is successfully created, the build process starts. Depending on how long back in the past the profile starts, this can take a considerable amount of time. You can follow the build process by clicking 'Continue' or at any time later by selecting the 'Stat' tab of the new profile. On the top of the Status information you will see a progress bar, showing you the percentage of completion.


Home | Flows | Packets | Traffic | Details | Stats | Plugins

Building Profile: port80

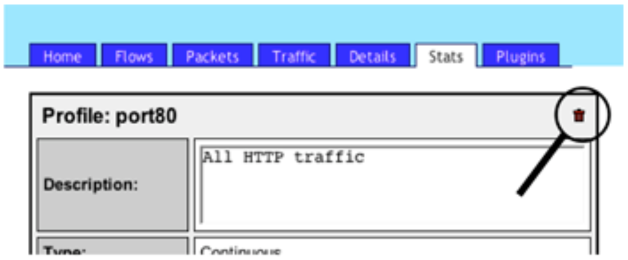
55.6%

Description:	All HTTP trafic
Type:	Continuous
Start:	2005-02-22-09-40

Progress of building the profile

2.3.5 Managing Profiles

Once a profile is created you can change the expire settings of a continuous profile. Select the '**Stat**' tab of the profile and click on the edit icon of the appropriate expire value. A continuous profile may expire due to the age of the data or the profile size used on disk. Expiring starts whenever one of the two limits is reached. Expiring ends at the configured value **$low_water** in the config file nfsen.conf.

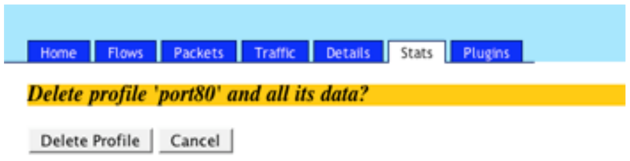

Home | Flows | Packets | Traffic | Details | Stats | Plugins

Profile: port80

Description:	All HTTP traffic

Delete Profile

To delete a profile, click on the trash can on the upper right corner of the profile info table. You will have to confirm to delete the profile:


Home | Flows | Packets | Traffic | Details | Stats | Plugins

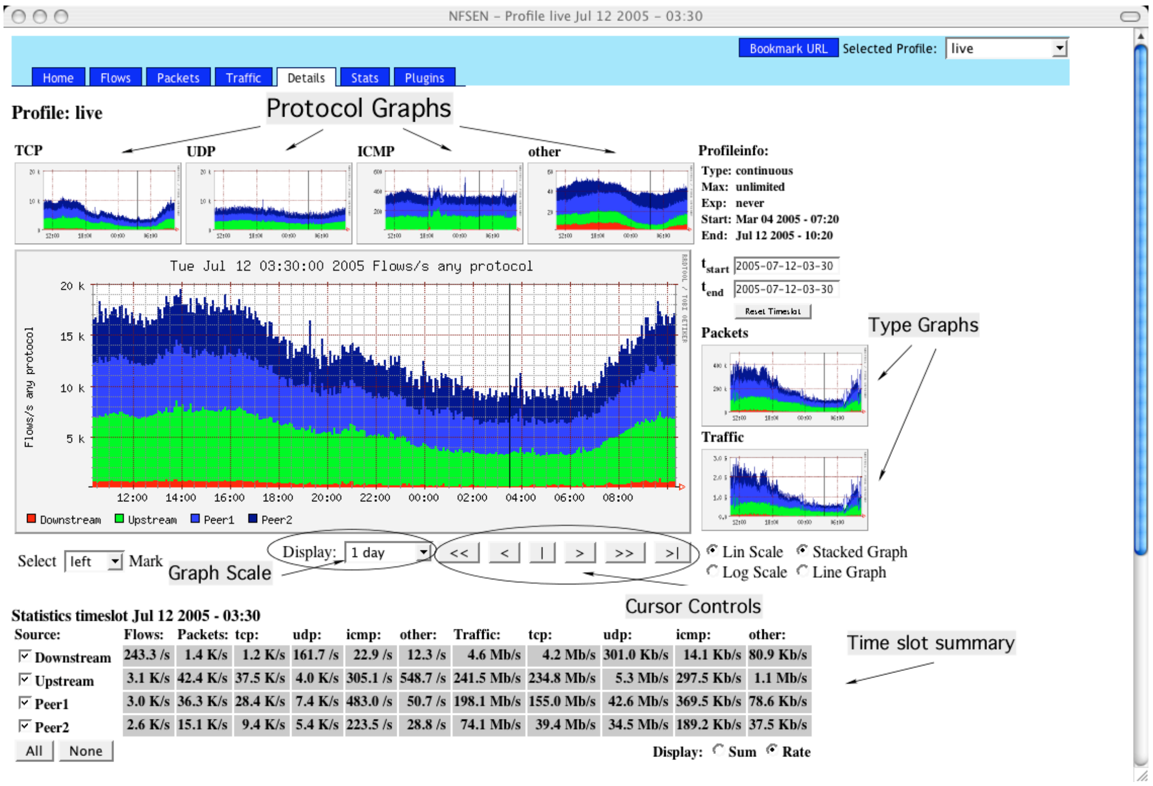
Delete profile 'port80' and all its data?

Delete Profile | Cancel

Confirm Delete Profile

2.3.6 Navigation

Detailed navigation and netflow processing is done in the 'Details' view. When entering this view, you will see the navigation display.



Profile: live

Profileinfo:
Type: continuous
Max: unlimited
Exp: never
Start: Mar 04 2005 - 07:20
End: Jul 12 2005 - 10:20

t_{start} 2005-07-12-03-30
t_{end} 2005-07-12-03-30

[✓ Lin Scale] Log Scale [✓ Stacked Graph] Line Graph

Statistics timeslot Jul 12 2005 - 03:30

Source:	Flows:	Packets:	tcp:	udp:	icmp:	other:	Traffic:	tcp:	udp:	icmp:	other:
✓ Downstream	243.3 /s	1.4 K/s	1.2 K/s	161.7 /s	22.9 /s	12.3 /s	4.6 Mb/s	4.2 Mb/s	301.0 Kb/s	14.1 Kb/s	80.9 Kb/s
✓ Upstream	3.1 K/s	42.4 K/s	37.5 K/s	4.0 K/s	305.1 /s	548.7 /s	241.5 Mb/s	234.8 Mb/s	5.3 Mb/s	297.5 Kb/s	1.1 Mb/s
✓ Peer1	3.0 K/s	36.3 K/s	28.4 K/s	7.4 K/s	483.0 /s	50.7 /s	198.1 Mb/s	155.0 Mb/s	42.6 Mb/s	369.5 Kb/s	78.6 Kb/s
✓ Peer2	2.6 K/s	15.1 K/s	9.4 K/s	5.4 K/s	223.5 /s	28.8 /s	74.1 Mb/s	39.4 Mb/s	34.5 Mb/s	189.2 Kb/s	37.5 Kb/s

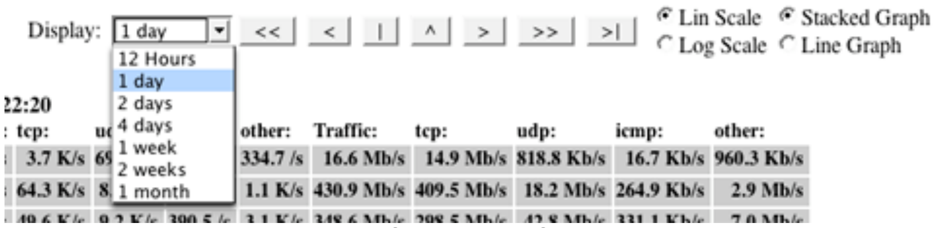
Display: Sum [✓ Rate]

Navigation Display

The page is divided into two parts: The upper part allows you to navigate through the netflow data as well as selecting a time slot or time window. The lower part contains all the controls to process the netflow data of the selected time slot or time window.

Clicking on any of the small protocol graphs will replace the main graphics with the selected protocol graph. You can switch back and forth and select the protocol for the main graph, which is appropriate for investigating your current situation. Clicking on the small type graphs on the right will replace the main graph as well as the protocol graphs with the selected type. Therefore you can switch to the 'Flows' 'Packets' or 'Bytes' graphics according your needs.

The time span of the graph can be changed using the pull down menu.



[✓ Lin Scale] Log Scale [✓ Stacked Graph] Line Graph

22:20

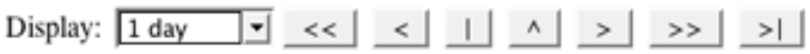
tcp:	u		other:	Traffic:	tcp:	udp:	icmp:	other:
3.7 K/s	69	[illegible]	334.7 /s	16.6 Mb/s	14.9 Mb/s	818.8 Kb/s	16.7 Kb/s	960.3 Kb/s
64.3 K/s	8	[illegible]	1.1 K/s	430.9 Mb/s	409.5 Mb/s	18.2 Mb/s	264.9 Kb/s	2.9 Mb/s

Select Time Span

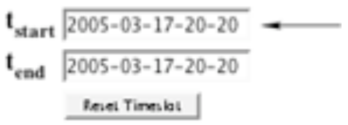
2.3.7 Selecting a time slot or a time window

A time slot starts at every 5 minutes cycle of the hour (0, 5, 10, 15 etc.) and lasts 5 minutes. A time window consists of several time slots. When entering the '**Details view**' a window scale of one days is selected so you will see the last 24 hours of the profile. The time cursor is placed in the middle of the begin and end of these 24 hours and the time window slot is set to one time slot. You will see the selected time slot or time window always in the title of the browser window as well as in the title of the main graph. Selecting a different time slot can be done in a number of ways:

- Clicking into the main graph, for example when you see a suspicious peak.
- Using the time cursor controls:

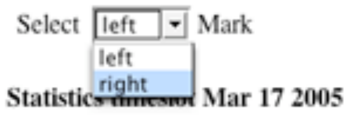


>	Next time slot: Advance time by 5 minutes.
<	Previous time slot: Go back 5 minutes.
>>	Advance time slot by a full time span of the graph.
<<	Go back by a full time span of the graph.
>|	Go to the end of the profile.
|	Center time cursor in current graph.
^	Place cursor at peak, found within +/- 1 hour time span of current cursor position.



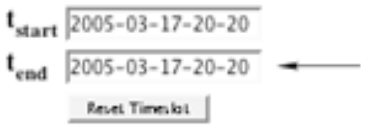
Entering the timeslot of interest in the t_{start} input field and press the enter key.

The graphs are immediately updated, when selecting a different time slot.
Sometimes it is desirable to process more than a single 5 min timeslot. The time range of interest an be selected by extending the time window. First, place the time cursor on the left edge of he requested time window, using the methods described above. Then, select the right edge of the time window by:



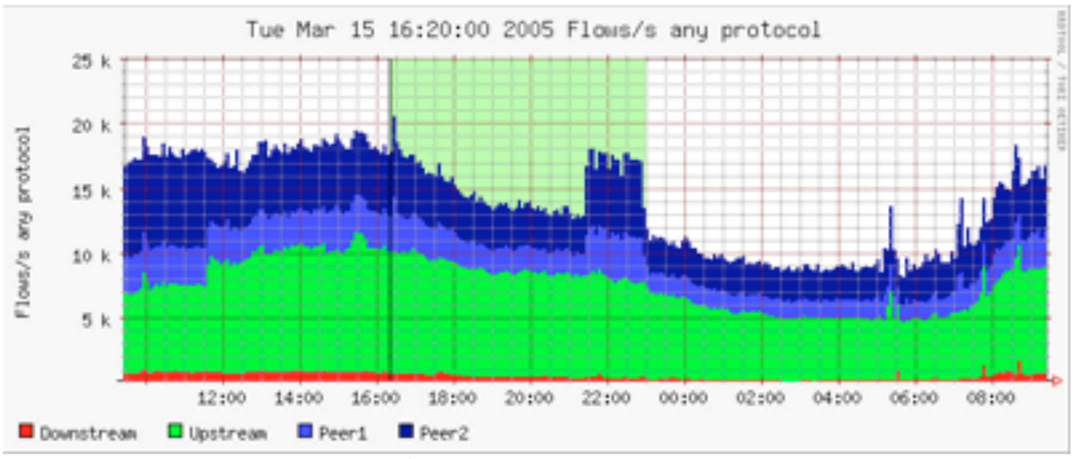
Select 'right' Mark from the pull down menu and click into the main graph.

or



Enter the right boundary in the t_{end} input field and press the enter key.

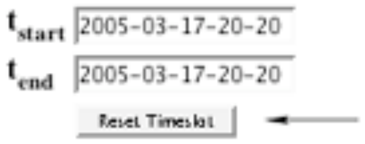
The main graph will be immediately updated with the selected time window:



Selected Time Window

Note:

- A selected time window may be shifted by selecting a new left boundary. The size of the time window remains.
- To adjust the window size, select a new right window boundary.



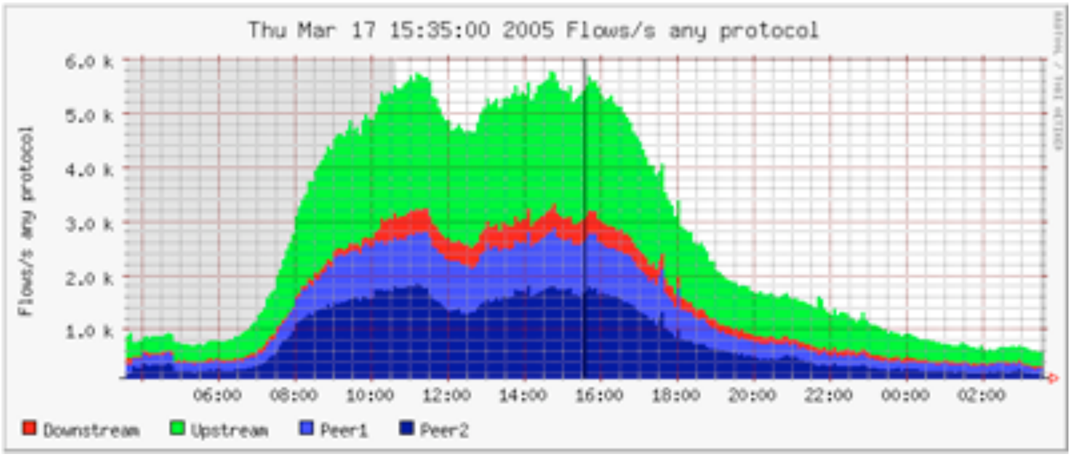
t_{start} 2005-03-17-20-20

t_{end} 2005-03-17-20-20

Reset Timeslot

To reset the window to the default size use the button below the t_{start}, t_{end} input boxes.

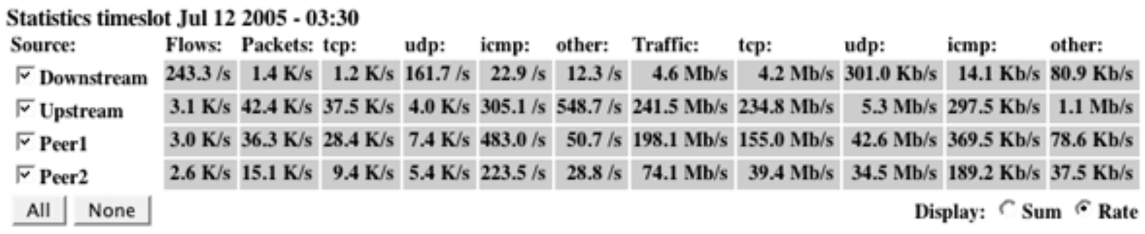
When you move the timeslot towards the begin of the profile, a greyed out area in the main graph appears and shows the end of the available netflow data. No data exists in the grey area. This border moves dynamically, when data expires. The run of the flows, packets and bytes graph may still be available, as this data is stored in the RRD database.



Border of available Data

2.3.8 Statistic Summary

The statistic summary below the main graph gives you an overview about **flows**, **packets** and **traffic** of the selected time slot or time window. The summary can be switched between the total sum of the selected time window, or the rate values per second.



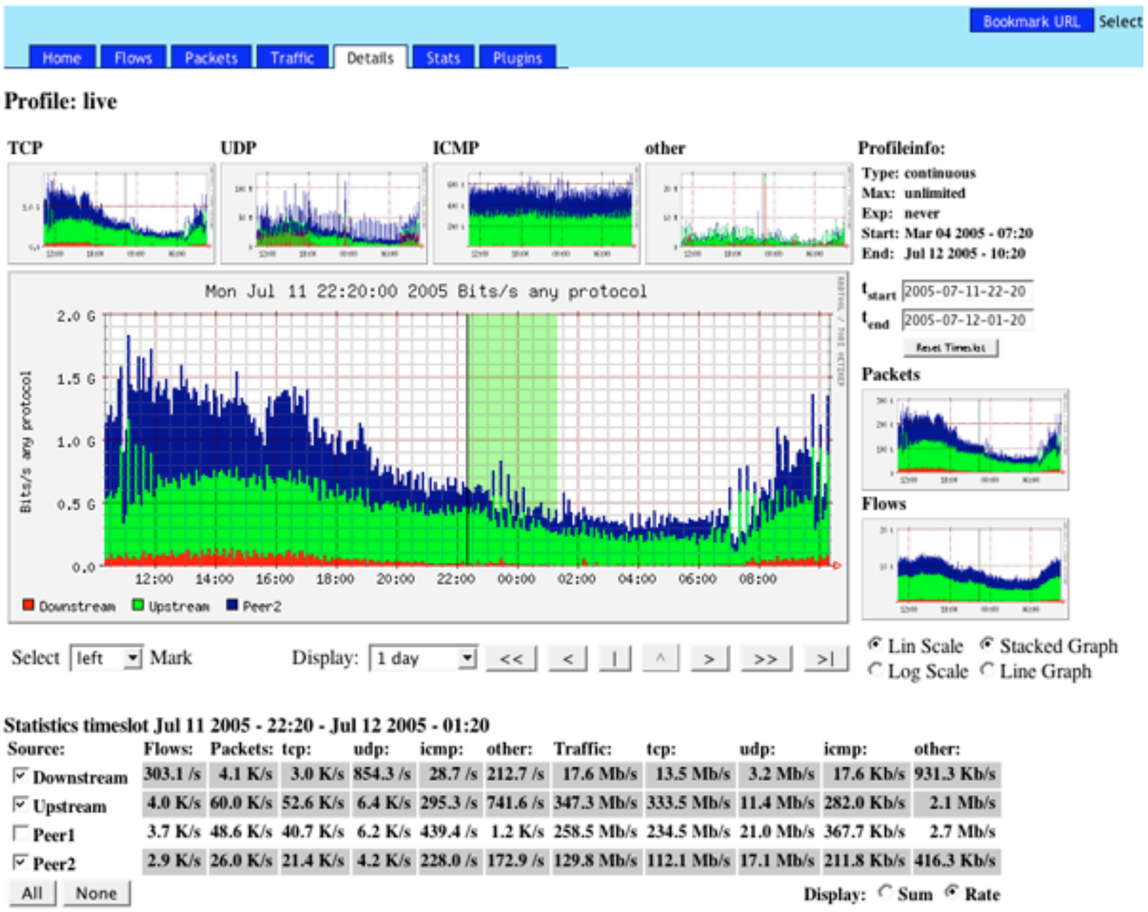
Statistics timeslot Jul 12 2005 - 03:30

Source:	Flows:	Packets:	tcp:	udp:	icmp:	other:	Traffic:	tcp:	udp:	icmp:	other:
✓ Downstream	243.3 /s	1.4 K/s	1.2 K/s	161.7 /s	22.9 /s	12.3 /s	4.6 Mb/s	4.2 Mb/s	301.0 Kb/s	14.1 Kb/s	80.9 Kb/s
✓ Upstream	3.1 K/s	42.4 K/s	37.5 K/s	4.0 K/s	305.1 /s	548.7 /s	241.5 Mb/s	234.8 Mb/s	5.3 Mb/s	297.5 Kb/s	1.1 Mb/s
✓ Peer1	3.0 K/s	36.3 K/s	28.4 K/s	7.4 K/s	483.0 /s	50.7 /s	198.1 Mb/s	155.0 Mb/s	42.6 Mb/s	369.5 Kb/s	78.6 Kb/s
✓ Peer2	2.6 K/s	15.1 K/s	9.4 K/s	5.4 K/s	223.5 /s	28.8 /s	74.1 Mb/s	39.4 Mb/s	34.5 Mb/s	189.2 Kb/s	37.5 Kb/s

All None

Display: Sum [✓ Rate]

Each line corresponds to one configured netflow source. If you are interested in only some of the sources, you may remove the others by clicking the checkboxes. This disables or enables this source in all graphs and in the statistics respectively.



Profile: live

Profileinfo:
Type: continuous
Max: unlimited
Exp: never
Start: Mar 04 2005 - 07:20
End: Jul 12 2005 - 10:20

t_{start} 2005-07-11-22-20
t_{end} 2005-07-12-01-20

Select: left, Mark
Display: 1 day
[✓ Lin Scale] Log Scale
[✓ Stacked Graph] Line Graph

Statistics timeslot Jul 11 2005 - 22:20 - Jul 12 2005 - 01:20

Source:	Flows:	Packets:	tcp:	udp:	icmp:	other:	Traffic:	tcp:	udp:	icmp:	other:
✓ Downstream	303.1 /s	4.1 K/s	3.0 K/s	854.3 /s	28.7 /s	212.7 /s	17.6 Mb/s	13.5 Mb/s	3.2 Mb/s	17.6 Kb/s	931.3 Kb/s
✓ Upstream	4.0 K/s	60.0 K/s	52.6 K/s	6.4 K/s	295.3 /s	741.6 /s	347.3 Mb/s	333.5 Mb/s	11.4 Mb/s	282.0 Kb/s	2.1 Mb/s
☐ Peer1	3.7 K/s	48.6 K/s	40.7 K/s	6.2 K/s	439.4 /s	1.2 K/s	258.5 Mb/s	234.5 Mb/s	21.0 Mb/s	367.7 Kb/s	2.7 Mb/s
✓ Peer2	2.9 K/s	26.0 K/s	21.4 K/s	4.2 K/s	228.0 /s	172.9 /s	129.8 Mb/s	112.1 Mb/s	17.1 Mb/s	211.8 Kb/s	416.3 Kb/s

Display: Sum [✓ Rate]

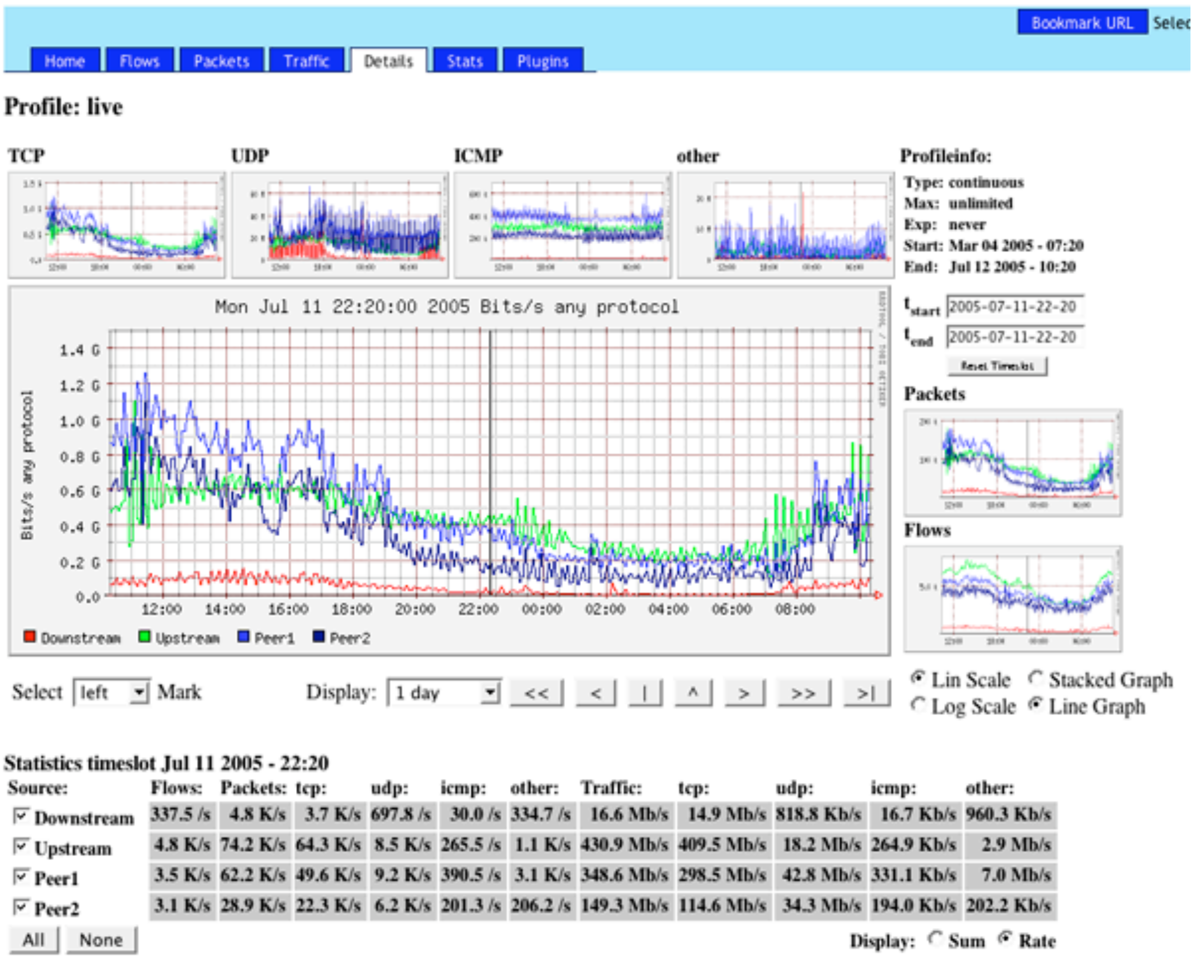
Disabled sources 'Downstream' and 'Peer1'
Enabling/disabling sources may rescale the graphs and you may get a more detailed graph and a different resolution on the y-axis.

2.3.9 Graph Display Options

To view the details, which your are interested in, a graph may be displayed with different options:

- Scale:
  - Linear y-axis
  - Logarithmic y-axis.
- Graph Type:
  - Stacked: All sources are drawn on top of each other.
  - Line: All sources are drawn independent.

Example of a line graph:



Profileinfo:
Type: continuous
Max: unlimited
Exp: never
Start: Mar 04 2005 - 07:20
End: Jul 12 2005 - 10:20

t_{start} 2005-07-11-22-20
t_{end} 2005-07-11-22-20

Select left Mark Display: 1 day

[✓ Lin Scale] Stacked Graph
Log Scale [✓ Line Graph]

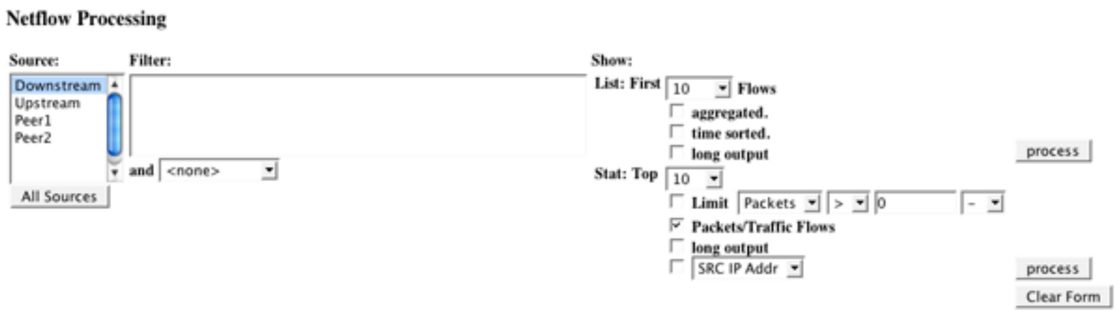
Statistics timeslot Jul 11 2005 - 22:20

Source:	Flows:	Packets:	tcp:	udp:	icmp:	other:	Traffic:	tcp:	udp:	icmp:	other:
✓ Downstream	337.5 /s	4.8 K/s	3.7 K/s	697.8 /s	30.0 /s	334.7 /s	16.6 Mb/s	14.9 Mb/s	818.8 Kb/s	16.7 Kb/s	960.3 Kb/s
✓ Upstream	4.8 K/s	74.2 K/s	64.3 K/s	8.5 K/s	265.5 /s	1.1 K/s	430.9 Mb/s	409.5 Mb/s	18.2 Mb/s	264.9 Kb/s	2.9 Mb/s
✓ Peer1	3.5 K/s	62.2 K/s	49.6 K/s	9.2 K/s	390.5 /s	3.1 K/s	348.6 Mb/s	298.5 Mb/s	42.8 Mb/s	331.1 Kb/s	7.0 Mb/s
✓ Peer2	3.1 K/s	28.9 K/s	22.3 K/s	6.2 K/s	201.3 /s	206.2 /s	149.3 Mb/s	114.6 Mb/s	34.3 Mb/s	194.0 Kb/s	202.2 Kb/s

Display: Sum [✓ Rate]

2.3.10 Netflow Processing

Once you have selected the time window of interest, you can process and filter the netflow data according your needs, using the process form.



List: First 10 Flows
☐ aggregated.
☐ time sorted.
☐ long output

Stat: Top 10
☐ Limit Packets > 0 -
☑ Packets/Traffic Flows
☐ long output
☐ SRC IP Addr

- Select the netflow sources to process. You may select multiple sources.
- Enter a netflow filter. The syntax conforms to the nfdump filter syntax.
- Select any options.

Click '**process**'.

The selected sources, the filter and all options from the processing form are compiled into the appropriate nfdump command. Therefore the filter syntax is exactly as described in 1.6. The time slot for processing is deviated from the time window selected in the main graph.

2.3.10.1 Options

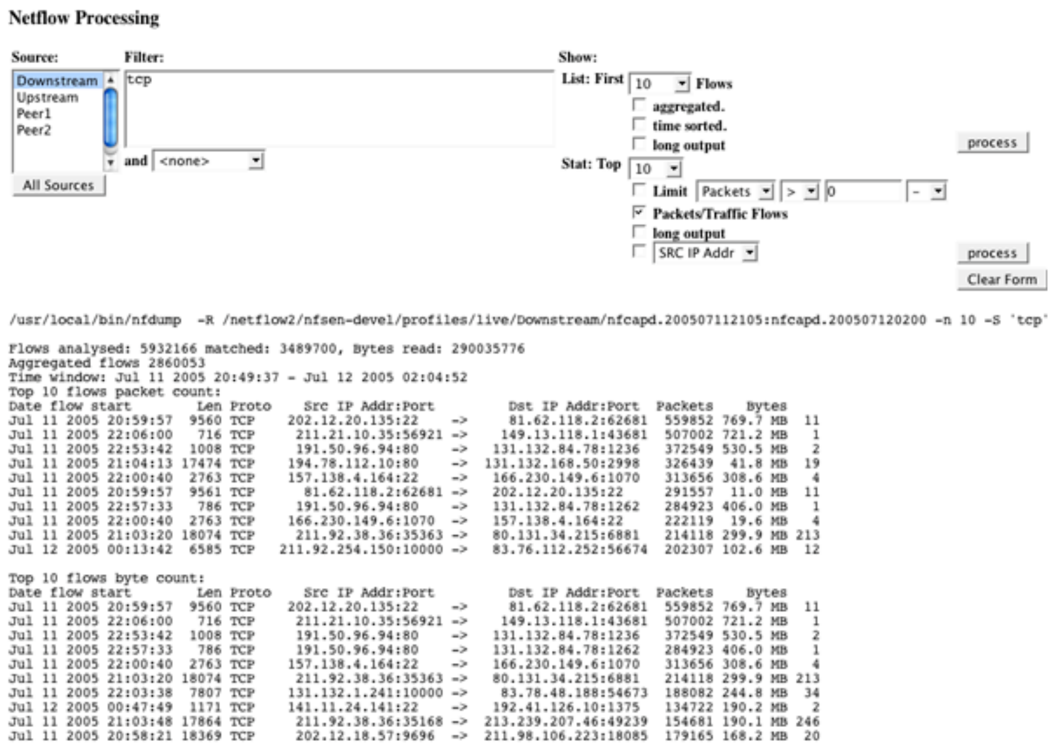
The list of the following options is used to compile the nfdump command. See also the nfdump man page for further details.

List Options

List First N Flows	List only the first N flows of the selected time slot nfdump option: -c N
aggregated	Aggregate the flows: SRC IP, DST IP, SRC Port, Dst Port nfdump option: -a
time sorted	Sort all flows according the time first seen, when reading data from multiple netflow sources. Important when tracking a certain period of time. nfdump option: -m
long output	Include TCP flags and tos field in output nfdump option: -o long

Statistic options

Stat Top N	Limit the statistics to the first N nfdump option: -n N
Limit Packets/Bytes	After creating the statistics, only show aggregated flow with more/less packets/flows than specified. nfdump option: +/-l num +/- L num
Packets/Bytes	Create the bytes and packets statistics nfdump option: -S
long output	Include TCP flags and tos field in output. nfdump option: -o long
SRC IP Addr.	Create statistics about selected parameter. nfdump option: -s <stat type>



Netflow Processing

Source:	Downstream (selected), Upstream, Peer1, Peer2
Filter:	tcp
and	<none>
Show: List: First	10 Flows
aggregated.	☐
time sorted.	☐
long output	☐
Stat: Top	10
Limit	☐ Packets > 0 -
Packets/Traffic Flows	✓
long output	☐
SRC IP Addr	☐

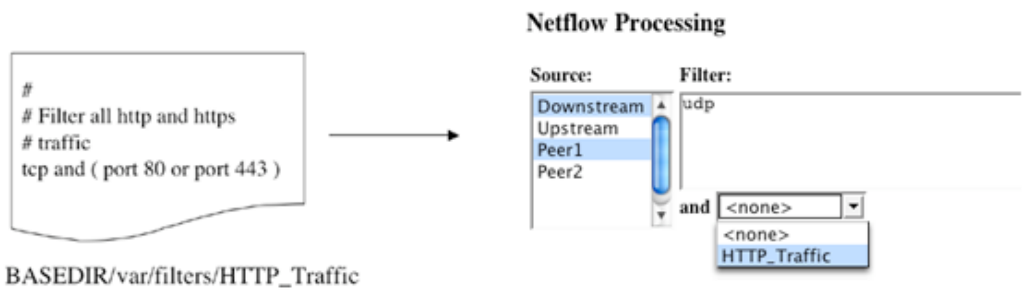
```
/usr/local/bin/nfdump  -R /netflow2/nfsen-devel/profiles/live/Downstream/nfcapd.200507112105:nfcapd.200507120200 -n 10 -S 'tcp'

Flows analysed: 5932166 matched: 3489700, Bytes read: 290035776
Aggregated flows 2860053
Time window: Jul 11 2005 20:49:37 - Jul 12 2005 02:04:52
Top 10 flows packet count:
Date flow start          Len Proto    Src IP Addr:Port            Dst IP Addr:Port  Packets    Bytes
Jul 11 2005 20:59:57  9560 TCP    202.12.20.135:22     ->      81.62.118.2:62681  559852 769.7 MB  11
Jul 11 2005 22:06:00   716 TCP     211.21.10.35:56921 ->     149.13.118.1:43681  507002 721.2 MB   1
Jul 11 2005 22:53:42  1008 TCP     191.50.96.94:80     ->   131.132.84.78:1236   372549 530.5 MB   2
Jul 11 2005 21:04:13 17474 TCP    194.78.112.10:80     ->  131.132.168.50:2998   326439  41.8 MB  19
Jul 11 2005 22:00:40  2763 TCP    157.138.4.164:22     ->   166.230.149.6:1070   313656 308.6 MB   4
Jul 11 2005 20:59:57  9561 TCP      81.62.118.2:62681 ->   202.12.20.135:22     291557  11.0 MB  11
Jul 11 2005 22:57:33   786 TCP     191.50.96.94:80     ->   131.132.84.78:1262   284923 406.0 MB   1
Jul 11 2005 22:00:40  2763 TCP    166.230.149.6:1070   ->   157.138.4.164:22     222119  19.6 MB   4
Jul 11 2005 21:03:20 18074 TCP     211.92.38.36:35363 ->   80.131.34.215:6881   214118 299.9 MB 213
Jul 12 2005 00:13:42  6585 TCP   211.92.254.150:10000 ->   83.76.112.252:56674  202307 102.6 MB  12

Top 10 flows byte count:
Date flow start          Len Proto    Src IP Addr:Port            Dst IP Addr:Port  Packets    Bytes
Jul 11 2005 20:59:57  9560 TCP    202.12.20.135:22     ->      81.62.118.2:62681  559852 769.7 MB  11
Jul 11 2005 22:06:00   716 TCP     211.21.10.35:56921 ->     149.13.118.1:43681  507002 721.2 MB   1
Jul 11 2005 22:53:42  1008 TCP     191.50.96.94:80     ->   131.132.84.78:1236   372549 530.5 MB   2
Jul 11 2005 22:57:33   786 TCP     191.50.96.94:80     ->   131.132.84.78:1262   284923 406.0 MB   1
Jul 11 2005 22:00:40  2763 TCP    157.138.4.164:22     ->   166.230.149.6:1070   313656 308.6 MB   4
Jul 11 2005 21:03:20 18074 TCP     211.92.38.36:35363 ->   80.131.34.215:6881   214118 299.9 MB 213
Jul 11 2005 22:03:38  7807 TCP   131.132.1.241:10000 ->     83.78.48.188:54673  188082 244.8 MB  34
Jul 12 2005 00:47:49  1171 TCP    141.11.24.141:22     ->   192.41.126.10:1375   134722 190.2 MB   2
Jul 11 2005 21:03:48 17864 TCP     211.92.38.36:35168 ->  213.239.207.46:49239  154681 190.1 MB 246
Jul 11 2005 20:58:21 18369 TCP     202.12.18.57:9696   ->  211.98.106.223:18085  179165 168.2 MB  20
```

Note:

Depending on the size of your network, netflow processing may consume a lot of time and resources, when you select a large time window and multiple resources.

2.3.10.2 Default Filters

Frequently used processing filters can be stored in a file under BASEDIR/var/filters. These filters will be available in the processing form. The name of the filter in the menu corresponds to the file name. The filters use the standard nfdump syntax.



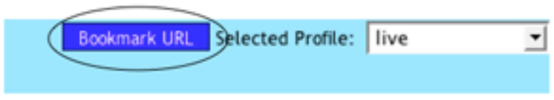
```
#
# Filter all http and https
# traffic
tcp and ( port 80 or port 443 )
```

BASEDIR/var/filters/HTTP_Traffic

The default filter is combined with the filter you enter in the text box. Both filters are linked with logical '**and**'. If a new installed filter is not visible in the menu after installing you may update the application cache by switching to the '**Stat**' tab and back again.

2.4 Bookmarks

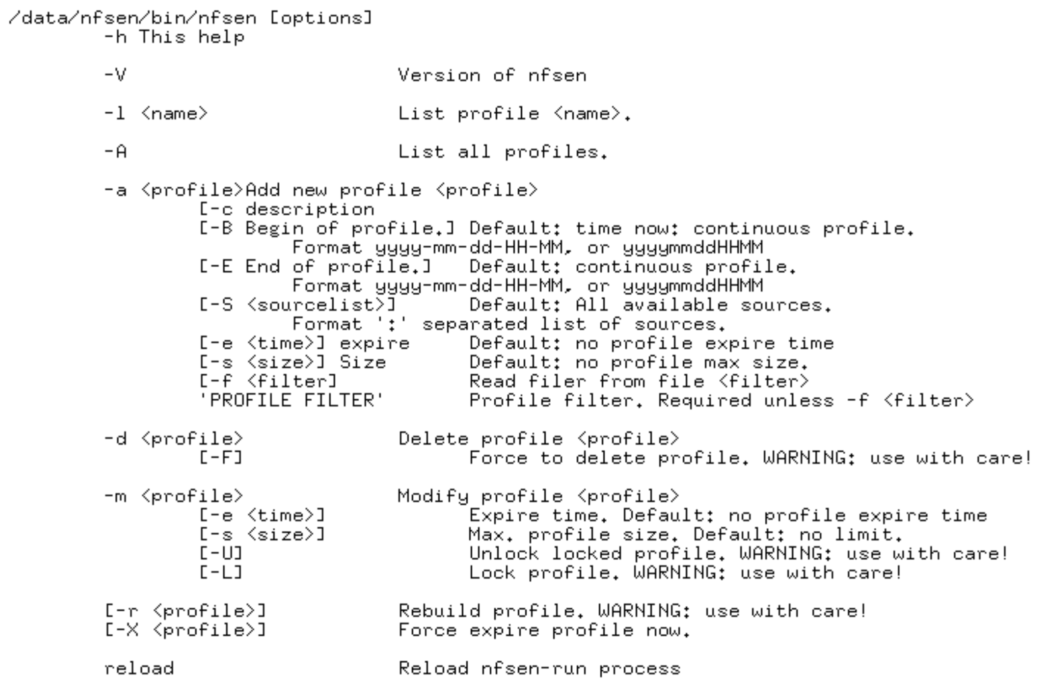
While working with NfSen, you may want to bookmark the current situation for later use or to send it as a link to a friend. The bookmark link at the top right of the page, allows you to do that.



Clicking on the link places the bookmark URL into the URL input field of your browser, allowing you to add this link to your bookmark collection. Many browsers also allow you 'right click' a link to copy the link location for pasting it in another application.

2.5 Command line tool 'nfsen'

The command line tool '**nfsen**' in the BASEDIR/bin directory works hand in hand with the frontend. It's used to create and manage profiles as you can do with the frontend in the '**Stat**' tab. Use **nfsen -h** to see all options available for **nfsen**:



```
/data/nfsen/bin/nfsen [options]
        -h This help

        -V                      Version of nfsen

        -l <name>               List profile <name>.

        -A                      List all profiles.

        -a <profile>Add new profile <profile>
                [-c description
                [-B Begin of profile.] Default: time now: continuous profile.
                        Format yyyy-mm-dd-HH-MM, or yyyymmddHHMM
                [-E End of profile.]   Default: continuous profile.
                        Format yyyy-mm-dd-HH-MM, or yyyymmddHHMM
                [-S <sourcelist>]      Default: All available sources.
                        Format ':' separated list of sources.
                [-e <time>] expire     Default: no profile expire time
                [-s <size>] Size       Default: no profile max size.
                [-f <filter]           Read filer from file <filter>
                'PROFILE FILTER'       Profile filter. Required unless -f <filter>

        -d <profile>            Delete profile <profile>
                [-F]                   Force to delete profile. WARNING: use with care!

        -m <profile>            Modify profile <profile>
                [-e <time>]            Expire time. Default: no profile expire time
                [-s <size>]            Max. profile size. Default: no limit.
                [-U]                   Unlock locked profile. WARNING: use with care!
                [-L]                   Lock profile. WARNING: use with care!

        [-r <profile>]          Rebuild profile. WARNING: use with care!
        [-X <profile>]          Force expire profile now.

        reload                  Reload nfsen-run process
```

If you create or delete a profile on the command line, the changes may not be instantly visible in the profile menu. Switching to the '**Stat**' tab updates the application cache and the profile menu.

3 Plugins

Even if NfSen is very flexible, you may have different or additional needs to process and display netflow data. This can be done using the plugin feature provided by NfSen. There are two type of plugins: **Backend** plugins and **Frontend** plugins.

The backend plugins are configured and installed as an extension to '**nfsen-run**', the background daemon, which keeps track with all the profile updates and data expiring. The plugins are Perl modules, which are loaded when NfSen is started or reloaded and are run at every 5 min interval, when nfcapd rolls over the data files. This allows you the process new netflow data as it becomes available and trigger any action of your choice. The backend plugin may store the output of the data processing, which in turn may be displayed with the frontend plugin. A frontend plugin is a simple php script, which is hooked into the web frontend and may display any results from the backend processing.

3.1 Installing Plugins

Plugins are stored in the BACKEND_PLUGINDIR and FRONTEND_PLUGINDIR respectively and are configured in nfsen.conf.
The configuration section is at the bottom of this file:

```
#
# plugins
# plugins are run for each timeslot, after the roll over of new data files.
# A plugin may run for any profile or for a specific profile only.
# Syntax: [ 'profile', 'module' ]
# profile: ',' separated list of profiles, or "*' for any profile
# module : Module name.
# The backend plugin is a Perl module and follows the standard
# Perl module conventions, with at least two additional functions: Init() and run().
# See demoplugin.pm for a simple template.
# The frontend plugin is a PHP script with dedicated functions
# <modulename>__ParseInput(), as well as <modulename_Run();

@plugins = (
    # profile # module
    [ 'live', 'TrackStats' ],
        );
```

Once your modules are installed and configured, signal the '**nfsen-run**' daemon to integrate the new plugin:

```
    BASEDIR/bin/nfsen reload
```

Have a look at the syslog file for errors when loading the plugins.

3.2 Writing Backend Plugins

Writing backend plugins is as easy as writing Perl modules. The template for a plugin may at least look like the example below:

```
#
#
package PluginName;

use strict;
use NfConf;

sub run {
  my $profile = shift;
  my $timeslot = shift; # Format: yyyymmddHHMM

  # Do whatever you want to do.
}

sub Init {
  return 1;
}

1;
```

The module has to provide at least two functions: **Init**() and **run**(). Init() is called, when the plugin is loaded. You may do any plugin specific initialization work. Return 1 for a successful initialization and to enable your plugin. Returning 0 disables your plugin and prevents the plugin from running.

The run() function is called periodically every 5 min, when new data becomes available. The first parameter specifies the profile name, the second parameter the new timeslot in the format 'yyyymmddHHMM'. Profile specific information can be retrieved using the NfSen.pm and NfConf.pm modules.

Example plugin TrackStats.pm: The example plugin below does a top 10 statistics every 5 minute and stores the result in a file for displaying it using the fronend plugin. Optionally, the result can be sent by email, if $NOTIFY = 1

```
#
package TrackStats;

use strict;
use NfSen;
use NfConf;

#
# The plugin may send any messages to syslog
# Do not initialize syslog, as this is done by
# the main process nfsen-run
use Sys::Syslog;
Sys::Syslog::setlogsock('unix');

# Use the optional Notification module
use Notification;

my ( $nfdump, $PROFILEDATADIR, $LOGFILE, $NOTIFY );

#
# Define a nice filter:
# We like to see flows from our network only
my $nf_filter = 'src net 172.16/16';

#
# Periodic function
#   input:  profile name
#          timeslot. Format yyyymmddHHMM e.g. 200503031200
sub run {
    my $profile  = shift;
    my $timeslot = shift;

    syslog('debug',"TrackStats run: Profile: $profile, Time: $timeslot");

    my %profileinfo     = NfSen::ReadProfile($profile);
    my $netflow_sources ="$PROFILEDATADIR/$profile/$profileinfo{'sourcelist'}";

    #
    # process all sources of this profile at once
    my @output = `$nfdump -M $netflow_sources -r nfcapd.$timeslot -s srcip '$nf_filter'`;

    #
    # Process the output and notify the duty team

    if (open (LOG, ">> $LOGFILE")){ ;
        print LOG @output ;
        close LOG ;
    } else {
        syslog('debug', "TrackStats: unable to open $LOGFILE") ;
    }
    if ( $NOTIFY ) {
        notify("TrackStats: Profile $profile, Timeslot $timeslot", \@output);
        syslog('debug', "TrackStats notify: ");
    }
} # End of run

sub Init {
    syslog("info", "TrackStats: Init");

    # Init some vars
    $nfdump         = "$NfConf::PREFIX/nfdump";
    $PROFILEDATADIR = "$NfConf::PROFILEDATADIR";
    $LOGFILE        = "$NfConf::VARDIR/tmp/trackstats.log" ;
    $NOTIFY         = 1 ;

    return 1;
} # End of Init

sub BEGIN {
    syslog("info", "TrackStats BEGIN");
    # Standard BEGIN Perl function - See Perl documentation
    # not used here
}

sub END {
    syslog("info", "TrackStats END");
    # Standard END Perl function - See Perl documentation
    # not used here
}

1;
```

3.3 Testing Backend Plugins

Before installing, test the plugin with the **testPlugin** script, available in BASEDIR/bin. The test scripts allows you to test the plugin with any available profile and time slot:

```
./testPlugin -p <pluginname> -P <profile> -t <timeslot>
```

3.4 Writing Frontend Plugins

A frontend plugin is a simple php script stored into the FRONTEND_PLUGINDIR. The script must have two well defined functions.

```
<?php

function <plugin_name>_ParseInput( $plugin_id ) {
    …
    // Set possible warning or error message
    $_SESSION['warning'] = "Warning set by plugin!";
    $_SESSION['error']   = "Error set by plugin!";

}
function <plugin_name>_Run( $plugin_id ) {

}

?>
```

The plugin_id is a unique integer, assigned to your plugin, which can be used for any purpose. Most often, it is used to identify input variables from html forms, belonging to this plugin.

Example of an possible input field definition in a html form:

```
<input type='text' name='<? echo "${plugin_id}_port";?>' value='' size='5' maxlength='5' >
```

This allows to distinguish between different plugins and prevents clashes with variable names in web forms.

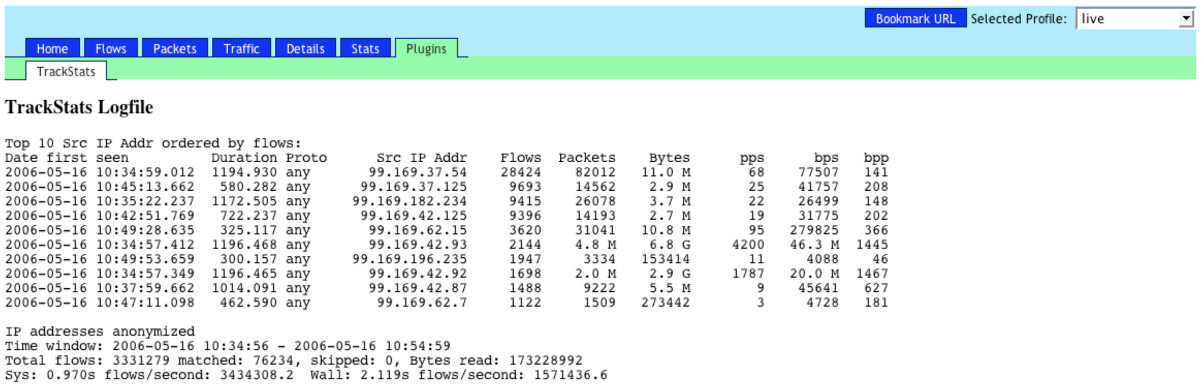
The **`<plugin_name>_ParseInput`** function is called, when your plugin is run, but before the NfSen navigation bar is sent. This allows you to parse any input fields from forms, and set the **`$_SESSION['warning']`** and **`$_SESSION['error']`** variables, in case of an error or warning has to be displayed.

The **`<plugin_name>_Run`** function is called after the navigation bar has been sent, and it's now up to the plugin to send more content to the client.

Example plugin TrackStats.php: The plugin below shows the result from the corresponding backend plugin TrackStats.pm described above.

```
<?php

/*
 * TrackStats plugin
 */

// Required functions
/*
 * This function is called prior to any output to the web browser and is intended
 * for the plugin to parse possible form data. This function is called only, if this
 * plugin is selected in the plugins tab
 */
function TrackStats_ParseInput( $plugin_id ) {

    /*
     * In TrackStats we have no input to parse, but this function must
     * exists anyway
     */

} // End of TrackStats_ParseInput


/*
 * This function is called after the header with the navigation bar have been
 * sent to the browser. It's now up to this function what to display.
 * This function is called only, if this plugin is selected in the plugins tab
 */
function TrackStats_Run( $plugin_id ) {

    global $VARDIR;

        if ($_SESSION['profile'] == 'live') {
            print "<h3>TrackStats Logfile</h3>\n";
            echo "<pre>";
            $logfile = "$VARDIR/tmp/trackstats.log";

            $lines = file($logfile) ;

            foreach ($lines as $line) {
                    echo htmlspecialchars($line) ;
            }
            echo "</pre>";
        } else {
                print "<h3>plugin not applicable for profile: " .$_SESSION['profile'] . "</h3>" ;
        }
} // End of TrackStats_Run

?>
```

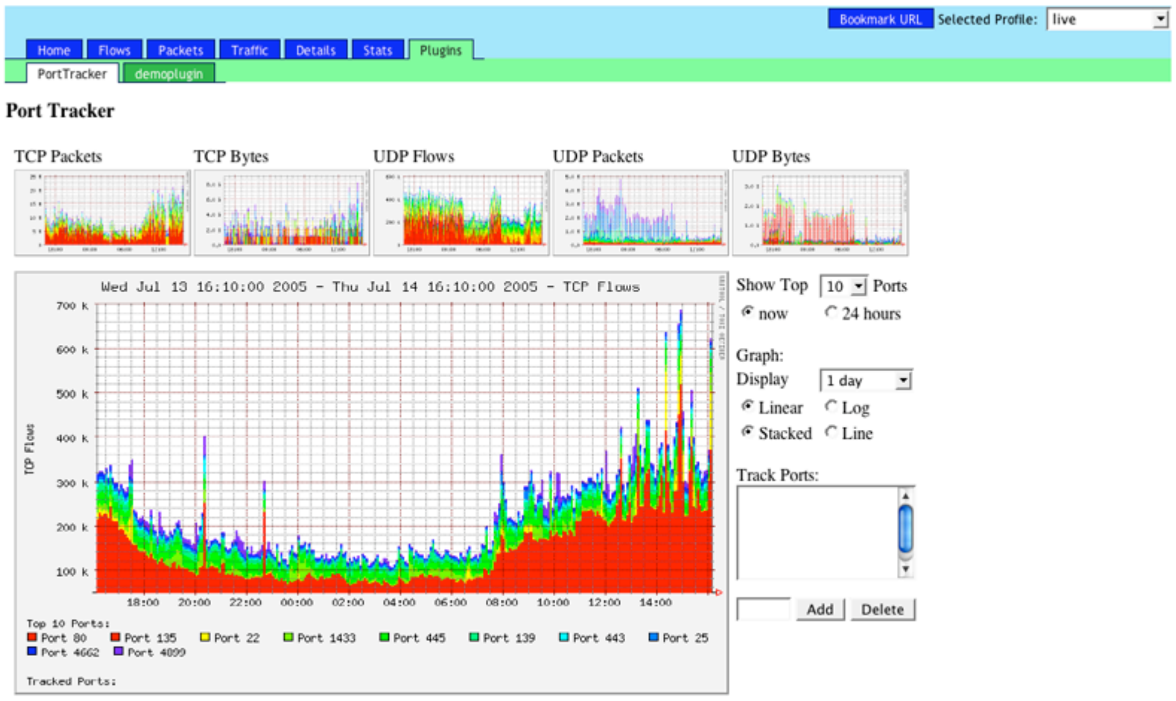
The frontend plugin is selected by the appropriate tab in the toolbar:



TrackStats Logfile

```
Top 10 Src IP Addr ordered by flows:
Date first seen          Duration Proto      Src IP Addr    Flows  Packets    Bytes      pps      bps   bpp
2006-05-16 10:34:59.012  1194.930 any       99.169.37.54    28424    82012   11.0 M       68    77507   141
2006-05-16 10:45:13.662   580.282 any      99.169.37.125     9693    14562    2.9 M       25    41757   208
2006-05-16 10:35:22.237  1172.505 any     99.169.182.234     9415    26078    3.7 M       22    26499   148
2006-05-16 10:42:51.769   722.237 any      99.169.42.125     9396    14193    2.7 M       19    31775   202
2006-05-16 10:49:28.635   325.117 any       99.169.62.15     3620    31041   10.8 M       95   279825   366
2006-05-16 10:34:57.412  1196.468 any       99.169.42.93     2144    4.8 M    6.8 G     4200   46.3 M  1445
2006-05-16 10:49:53.659   300.157 any     99.169.196.235     1947     3334   153414       11     4088    46
2006-05-16 10:34:57.349  1196.465 any       99.169.42.92     1698    2.0 M    2.9 G     1787   20.0 M  1467
2006-05-16 10:37:59.662  1014.091 any       99.169.42.87     1488     9222    5.5 M        9    45641   627
2006-05-16 10:47:11.098   462.590 any        99.169.62.7     1122     1509   273442        3     4728   181

IP addresses anonymized
Time window: 2006-05-16 10:34:56 - 2006-05-16 10:54:59
Total flows: 3331279 matched: 76234, skipped: 0, Bytes read: 173228992
Sys: 0.970s flows/second: 3434308.2  Wall: 2.119s flows/second: 1571436.6
```

As a more complex example – The experimental port tracker plugin:



3.5 Optional Modules

Optional modules are not required by NfSen. However, they make your life easier when writing plugins.

3.5.1 Notification.pm

Plugins may produce some output to be sent to a duty team for further analysis or for requesting some actions. This module allows you to send emails with one single line: **notify(<Subject>, <BODY>);** The configuration parameters, such as **From**, **To** and S**MTP server** are defined in the master nfsen.conf file.

Example:

```
nfsen.conf:
....
#
# Notification module
# The Notification module is an optional module. If you want your plugins to
# notify any result by email, use this module.
# Make sure you have installed Mail::Internet before using the module
#
# Use this from address
$MAIL_FROM = 'your@from.example.net' ;

# Use these recipients
$RCPT_TO = 'any@example.net', another@example.net" ;

# Use this SMTP server
$SMTP_SERVER = 'localhost' ;


Somewhere in your plugin:

use Notification;
...

@output = some command;
notify("Suspicious Flows found", \@output);
```