fd_filter & fd_extract

NeTraMet flow data file utility programs

Version 4.2

Nevil Brownlee

Information Technology Systems & Services
The University of Auckland
Auckland, New Zealand

September 1998

1. Introduction

fd_extract and fd_filter are programs for processing flow data files. They are intended first as utilities which perform basic utility functions on these files, and second as examples of code for working with them.

1.1. Overview

fd_filter reads a flow data file and processes it as requested by a format file so as to produce a new flow data file. Possible processing operations are:

- Compute flow rates, i.e. compute differences between samples for To/From Octet and PDU rates.
- Change the file format, e.g. rearrange the order of attributes in the flow data records.
- Filter out flows, i.e. select only those of interest. Identify these with a new 'tag' attribute.
- Filter out NeTraMet statistics records.

Any combination of these operations can be carried out by a single run of fd_filter.

fd_extract reads a flow data file and produces from it a file which is a column list of a matrix. The matrix has one row for each sample in the flow data, and one column for each specified set of flows. Such a file can be used as input to other utility programs, for example it can be plotted using gnuplot.

Version 3.1 added the ability to process the first sample from a 'trailer' file. This is most useful where the NeMaC 'flag file' has been used to close off a flow data file - e.g. at the end of a day - and start a new one. In such a case the first sample in the new file should also be the last sample in the old one. The 'trailer' option reads the header and format records of trailer file, checks that they match those of the input file, then reads the first sample of the trailer file as though it were concatenated with the input file.

Both programs are Unix filter programs, i.e. they read from an input file (usually standard input) and write to standard output. Instructions describing the processing operations are read from a format file.

1.2. Format Files

The syntax for format files is given below in the form of railway diagrams, and detailed examples are given in the following sections.

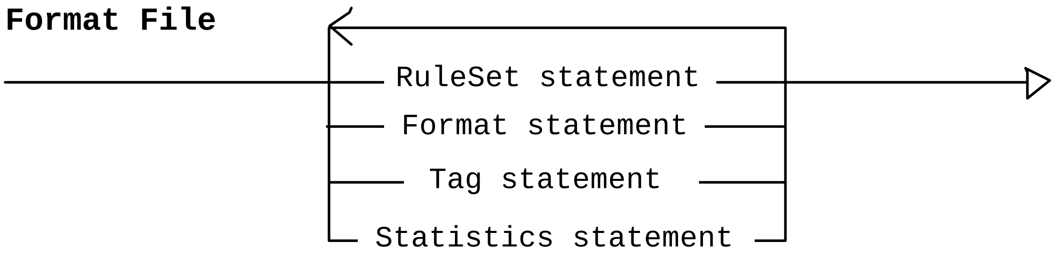
Each *statement* in a rule file starts at the beginning of a line and ends with a semicolon. A cross-hatch character marks the end of a line; all characters following a cross-hatch on a line are ignored by the scanner.

The scanner looks for keywords, numbers and addresses. Keywords are shown in the railway diagrams in upper case, but case is ignored by the scanner. Keywords, including attribute names, must be given in full – abbreviations are not allowed.

2. fd_filter

2.1. fd_filter Format File

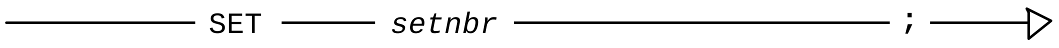
Format File



A filter format file contains one or more of four possible elements, which may appear in any order in the file. The format file *must* contain one format statement, the other statements are optional. The case of characters is not significant within rule files.

2.1.1. RuleSet Statement

RuleSet statement



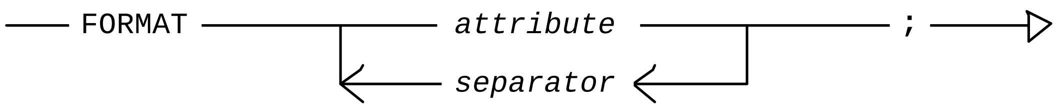
The RuleSet statement specifies the rule set from which flows are to be drawn. The default is not to filter out any rule sets, i.e. to copy all flows.

In versions 2 and 3 the rule set number in flow data records was taken directly from the SET statement in the rule file.

From Version 4.1, it identifies the rule set's index in the meters table of rule sets, and is chosen randomly by NeMaC. Version 4.1 flow data files include #Ruleset records which can be used by Analysis Applications to map the file's rule set numbers back to their original SET statement numbers. This is not yet implemented in fd_filter; experience has shown that it is sledom useful to filter on rule set number.

2.1.2. Format Statement

Format statement



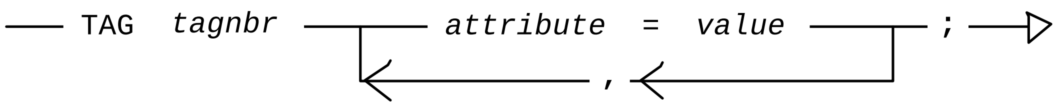
The Format statement specifies the format of rule data lines in the file written by fd_filter.

It starts with the FORMAT keyword, which is followed by a list of flow attributes in the order they are to appear in the Flow Data file. The attributes include all of the flow attributes described in the NeTraMet manual, and also the following:

ToOctetRate	Number of octets sent since previous collection
FromOctetRate	Number of octets received since previous collection
ToPDURate	Number of PDUs sent since previous collection
FromPDURate	Number of PDUs received since previous collection
TagNbr	Tag number for this flow (see 'Tag Statement' below)

2.1.3. Tag Statement

Tag statement



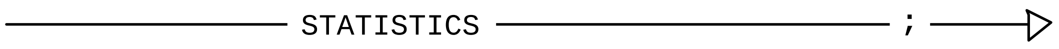
A tag statement identifies a flow as having a particular set of attribute values, and provides the tag as a convenient way to refer to those flows. For example we might use tag 2 to mean 'all flows with source address 130.216.3.1.' Note that there may be many flows which meet such a criterion. Using the tag in this way is much more convenient than using the {rule set number, flow number, start time} triple which the meter uses to uniquely define each flow, and which the user has no control over.

A tag statement starts with the keyword TAG, followed by a list of attribute values. The attributes allowed here are only those which are known to the meter, i.e. those documented in the NeTraMet manual. Attribute values are separated by commas, and the list is terminated by a semicolon.

If the format file contains no tag statements, all flows are copied to the output file.

2.1.4. Statistics Statement

Statistics statement



The Statistics statement tells fd_filter to copy meter performance statistics from the input flow data file.

2.2. Using fd_filter

To run fd_filter, give the following command:

```
fd_filter options format_file input_flow_file trailer_file
    > output_flow_file
```

This is the full version of the command, requesting fd_filter to read an input flow file followed by the first sample from a trailer flow file, process these as specified by a format file and write a new output flow file. If the input file isn't given fd_filter will read from standard input. fd_filter always writes to standard output, so the above command uses Unix output redirection to write to a disk file instead.

There are two options:

`-q`	supresses all 'commentary' output (which may be useful for daily jobs)
`-f`	prints the format lines from the data and format files

If no trailer file is specified fd_filter will stop at the end of the input flow data file, so information about flows created after the last sample may be lost. For example, if you were collecting 15-minute samples for Thursday and the last sample of the day were taken at 2345, Friday's file would start at 0000. Friday's first sample would contain data from all flows in the meter, including those which started after 2345 on Thursday. To filter out the set of flows active in the last quarter hour of Thursday, we must compare Thursday's last sample with Friday's first one. We would do this by executing fd_filter as follows:

```
fd_filter format_daily Thursday_flows Friday_flows > Thursday_traffic
```

2.3. A sample filter format file

Here is an example format file for fd_filter:

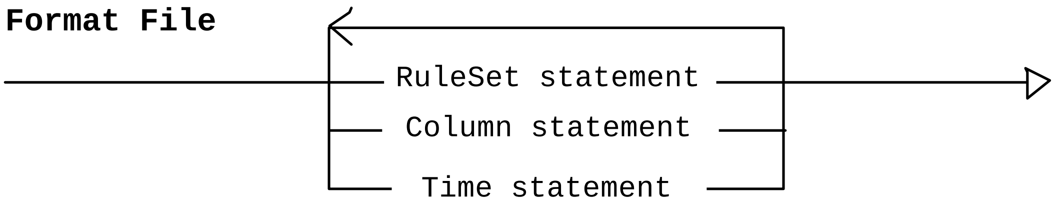
```
Format
   TagNbr SourcePeerType "\t"
   ToPduRate FromPduRate "\t" ToOctetRate FromOctetRate;
Tag 1
   SourcePeerType=IP;
Tag 2
   SourcePeerType=Novell;
Tag 3
   SourcePeerType=DECnet;
Tag 4
   SourcePeerType=EtherTalk;
Tag 5
   SourcePeerType=CLNS;
Tag 6
   SourcePeerType=Other;
```

This format file was written to read flow data files produced using the meter's default rule set, and produce an output file with Octet and PDU rates and tags for each of the network PeerTypes.

3. fd_extract

3.1. fd_extract Format File

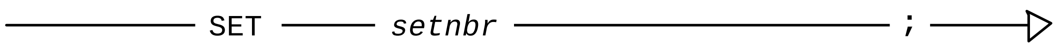
Format File



A filter format file contains one or more of three possible elements, which may appear in any order in the file. The case of characters is not significant within rule files.

3.1.1. RuleSet Statement

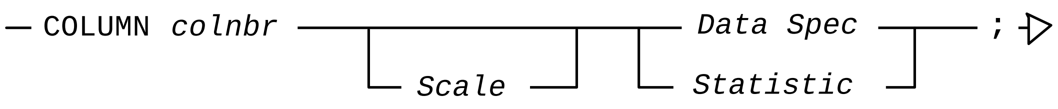
RuleSet statement



The RuleSet statement specifies the rule set from which flows are to be drawn. The default is not to filter out any rule sets, i.e. to copy all flows.

3.1.2. Column Statement

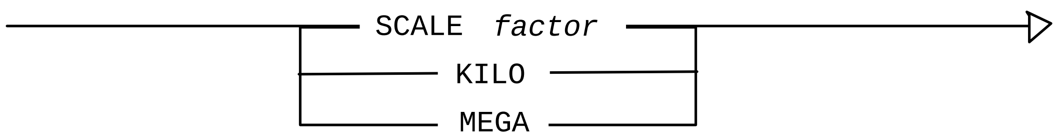
Column statement



The Column statement specifies one column of data to be written into the output file. It starts with the keyword COLUMN, followed by the column number. Column 1 is used for the sample time, so column numbers must start at 2. Be careful to specify each column once only!

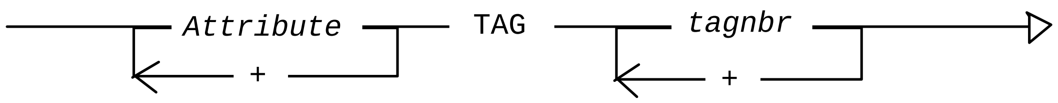
The Scale, if given, is a number used to divide the values from the flow data file before writing them to the output file. The default scale value is one.

Scale



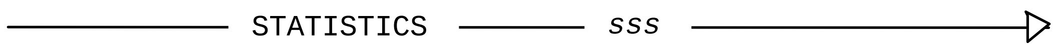
Any integer may be specified for the scale factor. KILO and MEGA provide a simple way to specify factors of 1,000 and 1,000,000 respectively.

Data Spec



The Data Spec tells fd_extract what data is to appear in the output column. It is given as a summation over attributes and tag numbers. For example we could request *ToOctetRate* + *FromOctetRate* for tags 3 + 5 + 8.

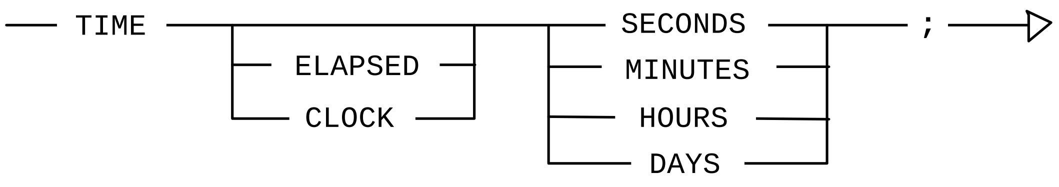
Statistic



The keyword STATISTICS indicates that one of the meter performance is to be used for this column instead of flow data. sss is the three-letter abbreviation for the particular statistic; these are listed in the NeTraMet manual.

3.1.3. Time Statement

Time Statement



The TIME statement specifies units for the time axis, which is always written to the output file as column 1. The time scale may be ELAPSED, i.e. the output time starts at 0 for the first output row, or CLOCK, in which case the output is the actual time of day. Time units may be SECONDS, MINUTES, HOURS or DAYS. The default time statement values are CLOCK HOURS.

3.2. Using fd_extract

To run fd_extract, give the following command:

```
fd_extract format_file input_flow_file > output_column_file
```

This is the full version of the command, requesting fd_extract to read an input flow file, process it as specified by a format file and write an output column-list file. If the input file isn't given fd_extract will read from standard input. fd_extract always writes to standard output, so the above command uses Unix output redirection to write to a disk file instead.

3.3. A sample extract format file

Here is an example format file for fd_extract:

```
time: elapsed minutes

column  2  scale 2000 ToOctetRate+FromOctetRate tag 1;  # IP
column  3  scale 2000 ToOctetRate+FromOctetRate tag 2;  # Novell
column  4  scale 2000 ToOctetRate+FromOctetRate tag 3;  # DECnet
column  5  scale 2000 ToOctetRate+FromOctetRate tag 4;  # EtherTalk
column  6  scale 2000 ToOctetRate+FromOctetRate tag 6;  # Other

column  7  scale 2    ToPDURate+FromPDURate tag 1;  # IP
column  8  scale 2    ToPDURate+FromPDURate tag 2;  # Novell
column  9  scale 2    ToPDURate+FromPDURate tag 3;  # DECnet
column 10  scale 2    ToPDURate+FromPDURate tag 4;  # EtherTalk
column 11  scale 2    ToPDURate+FromPDURate tag 6;  # Other
```

This format file was written to read flow data files produced by fd_filter using the example format file given above and produce an output plot file with elapsed minutes in column one, total OctetRates in columns 2 to 6 and total PDUrates in columns 7 to 11.