CHAPTER 4

Understanding the FlowCollector Data File Format

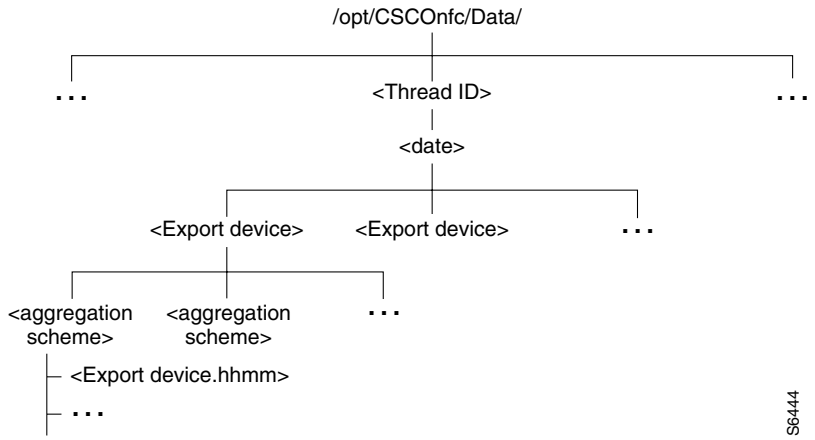
This chapter tells you how to interpret the data collected and saved in FlowCollector data files. The chapter includes information on the following topics:

Data File Directory Structure

Once you start FlowCollector, it begins to collect data based on your aggregation schemes and stores the collected data in data files.

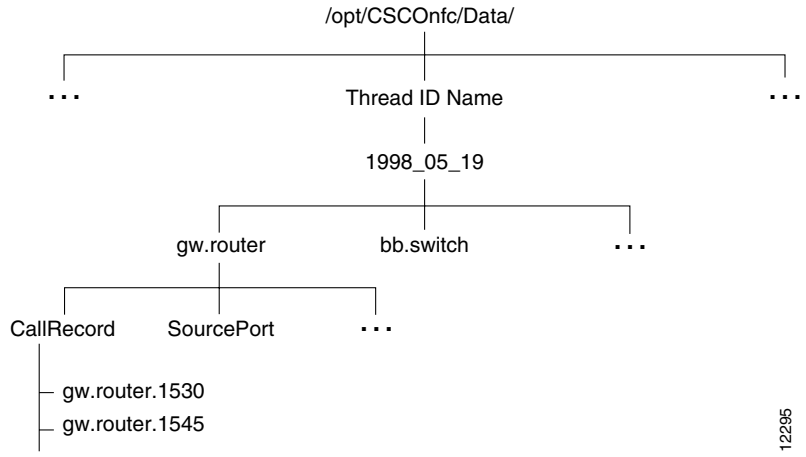
If you specified a custom data file directory path as the **DataSetPath** attribute for a thread, the data files are stored in the directory you specified. Otherwise, FlowCollector uses the default path, which is **/opt/CSCOnfc/Data**, and the default data file directory structure shown in Figure 4-1.

Figure 4-1 Default Data File Directory Structure



Starting with the root directory specified in the **DataSetPath** attribute of the thread (Figure 4-2 for an example), a directory is created that identifies the name of the Thread ID. A directory is created below that for each day (for example 1998_05_19). Under the date directory, a subdirectory is created for each export device (for example, **gw.router**) or **ROUTER_GROUPNAME** label, and under the export device, there is a subdirectory for each aggregation scheme (for example, **CallRecord** or **SourcePort**). The data files are stored by filename under the aggregation scheme subdirectory.

Figure 4-2 Data File Directory Structure Example



Caution Figure 4-1 and Figure 4-2 show the data file directory structure that is created in FlowCollector 3.6. However, this is not the default data file directory structure that is created after you first install FlowCollector 3.6. The Thread ID directory appears only if you set the **NFC20_COMPATIBLE_MODE** configuration parameter in the **nf.resources** file to **No** after installing FlowCollector 3.6.

See the "Understanding Installation Modes" section on page 2-2 for information about the default installation process, and see the "Modifying FlowCollector Resources" section on page 5-30 for information on how to reconfigure FlowCollector to obtain full FlowCollector 3.6 functionality.

For information on how filenames are formed, see the next section, Data Filenames.

Data Filenames

The name given to a data file takes either a long form (<*export-resource-name_yyyy_mm_dd.hhmm*>) or a short form (<*export-resource-name.hhmm*>), depending on the setting specified for the **LONG_OUTPUTFILE_SUFFIX** configuration parameter in the **nf.resources** file.

Note The short form of the name is the default file name. To use the long form, you must edit the **nf.resources** file and set the **LONG_OUTPUTFILE_SUFFIX** configuration parameter to **Yes**. For more information, see Table 5-6 on page 5-30.

Table 4-1 describes the fields of the data filename format.

Table 4-1 Data Filename Format Fields and Descriptions

Field	Description
export-resource-name	If this export device is named in the **ROUTER_GROUPNAME** configuration parameter, this value is the label defined by the **ROUTER_GROUPNAME** parameter; otherwise, the domain name system (DNS) name of the export device is the source of the data. If the DNS name is not available, the IP address of the export device is used. For information on the **ROUTER_GROUPNAME** configuration parameter, see the "Mapping a List of IP Addresses to One IP Address or Label" section on page 5-36.
hhmm	Time when the file was created in hours and minutes (local or Greenwich Mean Time [GMT]). FlowCollector uses GMT as the default time zone.
yyyy_mm_dd	Date in year, month, and day format.

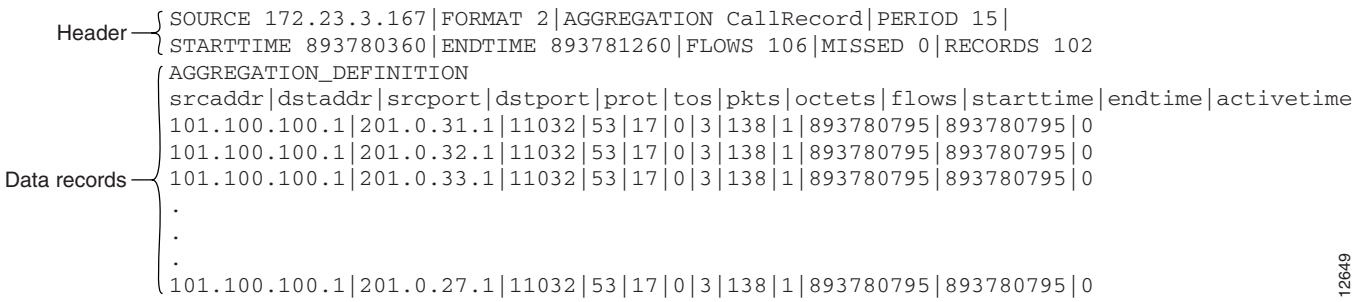
The following are examples of short and long data filenames:

```
gw-router.1530
gw-router_1996_03_15.1530
```

Data File Format

The format of the data file consists of the data file header and one or more aggregation definition data records. The **AGGREGATION_DEFINITION** section interprets the key and value fields in the data record area. With this information, data file users can determine the aggregation scheme used by parsing **AGGREGATION_DEFINITION**. The true order of the key and value fields is represented in the data file. The example in Figure 4-3 shows an abbreviated **CallRecord** data file.

Figure 4-3 Abbreviated CallRecord Data File



Caution Figure 4-3 shows an example of a **FORMAT 2** abbreviated data file created in FlowCollector 3.6. However, this is not the default data file format that is created after you first install FlowCollector 3.6. The **FORMAT A** data file type is created by default. This format does not include the **AGGREGATION_DEFINITION** section. This section appears only if you set the **NFC20_COMPATIBLE_MODE** configuration parameter in the **nf.resources** file to **No** after installing FlowCollector 3.6.

See the "Understanding Installation Modes" section on page 2-2 for information about the default installation process, and see the "Modifying FlowCollector Resources" section on page 5-30 for information on how to reconfigure FlowCollector to obtain full FlowCollector 3.6 functionality.

Data File Descriptions

FlowCollector data files are made up of two sections:

- Data File Header Section Format
- Aggregation Definition Format.

Data File Header Section Format

The data file header section consists of nine field-pairs, each made up of a keyword (all caps) and its corresponding value (*italic*) located to the right of the keyword. See Table 4-2 for descriptions of the fields in the data file header section.

```
SOURCE source|FORMAT format|AGGREGATION aggregation|PERIOD period|
STARTTIME time|ENDTIME time|FLOWS flows|MISSED missed|RECORDS records
```

Note Keywords and their value pairs are separated by either a vertical bar (|) or a comma (,). You can choose between a vertical bar or comma by using the **CSV_FORMAT** parameter in the **nf.resources** file.

Table 4-2 Data File Header Keywords and Descriptions

Keyword	Description
SOURCE	The label that identifies the source of the NetFlow export traffic summarized in this data file. The label can be the IP address, in dotted decimal format, or an ASCII name. If you are using the **ROUTER_GROUPNAME** feature, the label is the group name specified in the **ROUTER_GROUPNAME** configuration parameter.
FORMAT	A tag to track the version of the data file generated by FlowCollector. FlowCollector 2.0 uses tag **A**. FlowCollector 3.6 uses tag **2**.
AGGREGATION	The name of a valid, predefined aggregation scheme used to create this data file. See the "Aggregation Schemes" section on page 5-11 for more information.
PERIOD	Data collection period, specified in minutes. Under some circumstances, FlowCollector might generate a data file before the current data collection period expires. In such a case, FlowCollector adds the keyword **PARTIAL** to the filename, and the **PERIOD** field in the header is identified as **PERIOD PARTIAL**. For more information on partial data files, see the "Partial Data Files" section on page 4-7.
STARTTIME	The time in Coordinated Universal Time (UTC) seconds when this data collection period began.
ENDTIME	The time in UTC seconds when this data collection period ended.
FLOWS	The total number of NetFlow export records that are aggregated in this data file.

Table 4-2 Data File Header Keywords and Descriptions (continued)

Keyword	Description
MISSED	The number of flow records that FlowCollector should have received but did not. The MISSED value is derived from the sequence numbers (where present) in each packet. If the only data aggregated into a data file is from a V1 NetFlow export datagram or a V7 NetFlow export datagram with shortcut mode turned on, the MISSED field in the header contains -1 as the value. If any data is aggregated from datagrams in which sequence numbers are available (V5 or V7 without shortcut mode), the MISSED field in the header contains the actual count of missed flow records, even when there is a mix of V1, V5, or V7 NetFlow export datagrams.
RECORDS	The count of the aggregation records present in this data file.

See Figure 4-3 for an example of a typical data file that includes both the header and aggregation definition data records. A file that describes the aggregation definitions is located at **$NFC_DIR/include/NFC_Aggregation.h**.

Aggregation Definition Format

The body of a data file consists of one or more aggregation definition data records. Each aggregation definition consists of a keyword portion-one or more key fields-and a value portion-one or more value fields. For example, in the aggregation definition for the **CallRecord** aggregation scheme, the keyword portion is the first six fields and consists of the following aggregation scheme keywords:

```
srcaddr|dstaddr|srcport|dstport|prot|tos
```

The value portion is the last six fields and consists of the following aggregation values:

```
pkts|octets|flows|starttime|endtime|activetime
```

See Table 4-3 for descriptions of aggregation definition keywords, and Table 4-4 for descriptions of aggregation definition values. See Figure 4-3 for an example of a typical data file that includes both the header and aggregation definition data records.

Note In the aggregation definition, as in the header, keywords and their value pairs are separated by either a vertical bar (|) or a comma (,). You can choose between a vertical bar or comma by using the **CSV_FORMAT** parameter in the **nf.resources** file. Depending on the aggregation scheme you select, the aggregation definition data record contains a different combination of fields than that shown in the **CallRecord** data record example.

Table 4-3 Data File Aggregations Definition Keyword Descriptions

Keyword	Description
srcaddr	Source IP address
dstaddr	Destination IP address
src_subnet	Source subnet
dst_subnet	Destination subnet
src_mask	Source subnet mask
dst_mask	Destination subnet mask

Table 4-3 Data File Aggregations Definition Keyword Descriptions (continued)

Keyword	Description
src_user_subnet	Source user subnet.
dst_user_subnet	Destination user subnet.
src_as	Source autonomous system.
dst_as	Destination autonomous system.
srcport	Source port.
dstport	Destination port.
prot	Protocol field.
protocol	Protocol (**srcport**, **dstport**, and **prot** lookup).
input	Input interface.
output	Output interface.
tos	Type of service.
nexthop	Next hop IP address.

Table 4-4 Data File Aggregation Definition Value Descriptions

Value	Description
pkts	Packets
octets	Octets
flows	Flow count
starttime	First flow stamp (UTC sec.)
endtime	Last flow stamp (UTC sec.)
activetime	Total active time (ms)

Data File Example

The following data file example of the **CallRecord** aggregation scheme shows the data file header and the first two aggregation definition data records.

```
SOURCE 192.1.134.7|FORMAT 2|AGGREGATION CallRecord|PERIOD 15|STARTTIME 881972378|
ENDTIME 881973278|FLOWS 59709|MISSED 0|RECORDS 2345
AGGREGATION_DEFINITION
srcaddr|dstaddr|srcport|dstport|prot|tos|pkts|octets|flows|starttime|endtime|activetime
171.69.1.17|172.23.34.36|2963|6000|6|114|2|176|1|768550628|768550628|0
171.69.1.23|171.69.25.133|2972|6500|17|0|3|172|1|768520516|768520520|4135
.
.
.
```

In the **CallRecord** aggregation scheme, the key portion of the data record is the first six fields and consists of the following aggregation fields:

```
srcaddr|dstaddr|srcport|dstport|prot|tos
```

For example, the first six fields in the second data record from the example above are:

```
171.69.1.23|171.69.25.133|2972|6500|17|0
```

where the fields are described as follows:

Field	Description
171.69.1.23	Source IP address (srcaddr).
171.69.25.133	Destination IP address (dstaddr).
2972	Source port (srcport).
6500	Destination port (dstport).
17	Protocol byte (prot).
0	Type of service (ToS).

The value portion is the last six fields and consists of the following aggregation values:

```
pkts|octets|flows|starttime|endtime|activetime
```

For example, the last six fields in the second data record from the example above are:

```
3|172|1|768520516|768520520|4135
```

where the fields are described as follows:

Field	Description
3	Packet count.
172	Octet count.
1	Flow count.
768520516	Time in UTC seconds of the first packet that is summarized in this record.
768520520	Time in UTC seconds of the last packet that is summarized in this record.
4135	Active time, in milliseconds; defined as the sum of individual active time calculations for all the flows summarized into the current record.

Note The *activetime* value can be 0 if one single-packet flow produces this record. In this case, the value of the flows field is 1, and the values for *starttime* and *endtime* are identical.

Partial Data Files

Under normal circumstances, FlowCollector generates a data file every *n* minutes, as specified by the **Period** attribute in the thread definition (see the "Creating a Thread" section on page 5-8). For example, if the **Period** attribute in a thread is set to 10 minutes, FlowCollector collects data for 10 minutes, writes that data into a data file, and then starts over in a new data collection period.

Under certain circumstances, FlowCollector might be forced to generate a data file before the current data collection period expires. Such a data file is called a partial data file (because it does not represent data collected for the entire defined collection period) and occurs for one of these reasons:

- FlowCollector is stopped
- FlowCollector detects that one of its counters is about to wrap
- The NFUI is used to modify a thread definition.

The data in a partial data file is valid data; the file just does not represent a full data collection period and is differentiated to prevent data statistics from being distorted by comparing data from full and partial periods.

If FlowCollector has data in its internal aggregation buffers when one of the preceding conditions occurs, it writes the data into one or more data files. Because the current data collection period has not expired, FlowCollector generates and marks the data files differently: the keyword **PARTIAL** is added to the data filename as a suffix, and the **PERIOD** field in the header is identified as **PERIOD PARTIAL**.

In the following two data file examples, the first example shows a complete data file, and the second example shows a partial data file (using a different aggregation scheme).

```
SOURCE gw.router|FORMAT 2|AGGREGATION Protocol|PERIOD 10|STARTTIME 923416099|
ENDTIME 923416699|FLOWS 2868330|MISSED 154590|RECORDS 1
AGGREGATION_DEFINITION protocol|pkts|octets|flows
ICMP|2868330|2868330|2868330

SOURCE gw.router|FORMAT 2|AGGREGATION DestPort|PERIOD PARTIAL|
STARTTIME 923419273|ENDTIME 923419607|FLOWS 4467930|MISSED 0|RECORDS 250
AGGREGATION_DEFINITION dstport|pkts|octets|flows
1|17872|2626340864|17872
2|17872|2626340864|17872
..
..
```

Using the filesready File to Track Data Files

FlowCollector periodically appends the absolute path names of data files that it has generated to a list in a log file named **filesready**. FlowCollector identifies the file with the time stamp *YYYY_MM_DD*, where *YYYY_MM_DD* represents the year, month, and day. The **filesready** file is located with the other log files in the **$NFC_DIR/logs** directory. There is one such file per **DataSetPath** setting, per day.

Typically, a client application reads this file every *n* minutes, processes it to determine the names of any newly added data files, and then retrieves those new data files.

After it finishes writing a new data file, FlowCollector appends the absolute path name of the new data file onto the list in the **filesready** file. If FlowCollector deletes some data files as instructed by the **FileRetain** setting in its thread definitions, it updates the corresponding **filesready** file. The **filesready** file contains a header that indicates the format version in use. The following example shows the header, contents, and organization of a typical **filesready** file.

Note The following example is from FlowCollector 3.6. However, this is not the default format that is created after you first install FlowCollector 3.6. If you are running in NFC 2.0-compatible mode, the output begins with data and does not include the format header. The **FORMAT 1** identifier line appears only if you set the **NFC20_COMPATIBLE_MODE** configuration parameter in the **nf.resources** file to **No** after installing FlowCollector 3.6.

See the "Understanding Installation Modes" section on page 2-2 for information about the default installation process, and see the "Modifying FlowCollector Resources" section on page 5-30 for information on how to reconfigure FlowCollector to obtain full FlowCollector 3.6 functionality.

```
FORMAT 1
/opt/CSCOnfc/Data/1998_02_11/171.71.34.79/Protocol/171.71.34.79.2135
/opt/CSCOnfc/Data/1998_02_11/171.71.34.79/DetailASMatrix/171.71.34.79.2136
/opt/CSCOnfc/Data/1998_02_11/171.71.34.79/Protocol/171.71.34.79.2136
/opt/CSCOnfc/Data/1998_02_11/171.71.34.79/DetailASMatrix/171.71.34.79.2137
/opt/CSCOnfc/Data/1998_02_11/171.71.34.79/Protocol/171.71.34.79.2137
/opt/CSCOnfc/Data/1998_02_11/171.71.34.79/DetailASMatrix/171.71.34.79.2138
/opt/CSCOnfc/Data/1998_02_11/171.71.34.79/Protocol/171.71.34.79.2138
/opt/CSCOnfc/Data/1998_02_11/171.71.34.79/DetailASMatrix/171.71.34.79.2139
/opt/CSCOnfc/Data/1998_02_11/171.71.34.79/Protocol/171.71.34.79.2139
```