Network Working Group
Internet-Draft
Expires: May 15, 2008

C. Sommer
F. Dressler
Univ. Erlangen
G. Muenz
Univ. Tuebingen

November 12, 2007

Mediator-Specific Extensions to IPFIX Protocol and Information Model
<draft-sommer-ipfix-mediator-ext-00.txt>

Status of this Memo

Copyright Notice

Abstract

IPFIX supports the concept of an Mediator, a device that receives, transforms, and exports data streams using IPFIX. One of the most important requirements is the reduction of the volume of IPFIX traffic by discarding and aggregating received information. This document introduces a number of extensions to the IPFIX Protocol and IPFIX Information Model that support the export of aggregated IPFIX data. In particular, techniques are introduced that optimize the transport of descriptive information. Thus, more information can be preserved in the transmission while further reducing both the number and the size of IPFIX messages. All the proposed extensions are directly applicable to the IPFIX Mediator but can be used in many different applications as well.

Table of Contents

1. Introduction

The IPFIX Mediator is intended to provide techniques and features to process IPFIX data in a Mediation Process. This process receives data streams using IPFIX. It can apply transformations or aggregation techniques and forward the resulting Flow information to an Exporting Process and, thus, to another IPFIX collector. Flow aggregation is one of the most challenging and important operations in high-bandwidth networks. The main idea is to reduce both the number and the size of IPFIX messages. This document introduces extensions to the IPFIX Protocol and IPFIX Information Model that support the export of aggregated IPFIX data. In particular, a new Template type is introduced and additional Information Elements are described. All these extensions allow and optimize the transport of descriptive information on aggregated IPFIX data. Thus, more information can be preserved in the transmission while further reducing both the number and the size of IPFIX messages. All the proposed extensions are directly applicable to the IPFIX Mediator but can be used in many different applications as well.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119]. Illustrations of abstract data types are written in Augmented Backus-Naur Form (ABNF), as specified in [RFC4234], extending the abstract data types defined in [I-D.ietf-ipfix-info].

2. Terminology

Apart from the basic terms as defined in [I-D.ietf-ipfix-protocol], the following terms are used within this document:

Compound Flow:
A Compound Flow is the result of an aggregation of one or more individual input Flows that matched an Aggregation Rule. It might, for example, contain the total count of all packets addressed to a common subnet that were observed within a given time interval.

Aggregation Rule:
An Aggregation Rule defines the properties of a Compound Flow and the contents of the corresponding Flow Records. Optionally, a set of filtering criteria MAY be specified in order to restrict the applicability of the rule to those Flows that show certain patterns.

3. Rich Template

[I-D.dressler-ipfix-aggregation] describes how pattern matching is used to restrict the applicability of an Aggregation Rule and how patterns define Common Properties of the resulting Compound Flows. In order to avoid the overhead of the repeated transmissions of all Common Properties (or their identifiers) in all Flow Records, a new Template Set, the "Rich Template Set" is introduced. This Template Set allows an Exporting Process to simultaneously declare and transmit Common Properties to a receiver.

The basic format of a Rich Template Set is shown in Figure 1. It is the same as that of a Template Set defined in [I-D.ietf-ipfix-protocol], except for a different Set ID. The format of individual Rich Template Records, however, differs from that of Template Records and is shown in Figure 2.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Set ID = 4           |          Length               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                     Rich Template Record 1                    |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                              ...                              |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                     Rich Template Record N                    |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Padding (opt)                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1: Rich Template Set Format

The Rich Template Set field definitions are as follows:

Set ID
   Type of this Template Set. A Set ID value of 4 is proposed for the Rich Template Set.

Length
Total length of this set in bytes, as defined in [I-D.ietf-ipfix-protocol].

Padding
OPTIONAL padding, as defined in [I-D.ietf-ipfix-protocol].

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |  Template ID (> 255)          |  Field Count                  |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |  Data Count                   |  Common Properties ID         |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    |                      Field 1 Specifier                        |
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    |                             ...                               |
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    |                      Field N Specifier                        |
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    |                       Data 1 Specifier                        |
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    |                             ...                               |
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    |                       Data M Specifier                        |
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    |                         Data 1 Value                          |
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    |                             ...                               |
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    |                         Data M Value                          |
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: Rich Template Record Format

The Rich Template Record field definitions are as follows:

Template ID
Template ID of this Rich Template Record. As defined in [I-D.ietf-ipfix-protocol], this value MUST be greater than 255.

Field Count
Number of regular fields that will be sent in subsequent Data Records using this Template, as defined in [I-D.ietf-ipfix-protocol].

Data Count
Number of fixed-value fields that will be sent in this Template.

Common Properties ID
Contains an identifier that can be referred to by commonPropertiesId Information Elements, as introduced in [I-D.ietf-ipfix-reducing-redundancy].

Field N Specifier
Information Element identifier, Field length and an Enterprise Number (if applicable) of field N. Refer to [I-D.ietf-ipfix-protocol] for more information on Field Specifiers.

Data M Specifier
Same as "Field N Specifier", but used for Common Properties of all Data Records of this Template. Together with Data M Value, a similar encoding like TLV (type-length-value) is achieved.

Data M Value
Bit representation of a Common Property as would be transmitted in a Data Record.

Table 1 illustrates the relationship between field modifiers and patterns on the one hand, and the resulting regular and fixed-value fields in the Rich Template on the other hand. It can be seen that the analyzer is able to deduce all instructions of the Aggregation Rule considering the structure of the Rich Template, except the combination "discard without pattern" that does not result in any field.

field modifier	pattern	field in Flow Record	fixed-value field in Rich Template
discard	no	N/A	N/A
discard	yes	N/A	yes, contains pattern
keep	no	yes	N/A
keep	yes	yes, if pattern specifies a range of values	yes, contains pattern
mask	no	yes, IP network address	N/A
mask	yes	yes, IP network address	yes, contains pattern

Table 1: Relation between field modifiers, Flow Records, and Rich Templates

Assume, for example, the concentrator was given the Aggregation Rule shown in Table 2.

IPFIX Field	Filtering	Aggregation
sourceIPv4Address	192.0.2.0/28	discard
destinatonTransportPort		keep
packetDeltaCount		aggregate

Table 2: Example Rule

Based on the Aggregation Rule, the concentrator would now first send a corresponding Rich Template Record as shown in Table 3.

Field	Value
Template ID	10001
Field Count	2
Data Count	2
Common Properties ID	0
Field 1 Type	Destination Port
Field 2 Type	Packets
Data 1 Type	Source IP Prefix
Data 2 Type	Source IP Mask
Data 1 Value	192.0.2.0
Data 2 Value	28

Table 3: Rich Template used

Assume further that the concentrator receives the Flow Records shown in Table 4.

Source IP	Source Port	Destination IP	Destination Port	Packets
192.0.2.1	64235	192.0.2.101	80	10
192.0.2.2	64236	192.0.2.102	110	10
192.0.2.3	64237	192.0.2.103	80	10
192.0.2.101	64238	192.0.2.1	80	10
192.0.2.102	64239	192.0.2.2	80	10

Table 4: Incoming Flows

The concentrator would then export Data Records of this type, which contain the Compound Flows resulting from aggregation. Note that the Flows' Common Property, having a source IP address in 192.0.2.0/28, was already transmitted in the Rich Template Record and is thus not included in Data Records. The exported Data Records, shown in Table 5, only contain the aggregated packet counts and the destination port, the latter being the only discriminating Flow Key property.

Destination Port	Packets
80	20
110	10

Table 5: Aggregated Flows

4. Abstract data type ipv4Network

Currently, the transport of IP network information as specified by IPFIX is done using separate fields for the network address and the corresponding mask. We propose a new abstract data type ipv4Network that represents the common notation of IP networks: address/mask.

The ipv4Network abstract data type extends the abstract data type ipv4Address to allow a concatenated unsigned8 specifying the prefix length. Alternatively, Information Elements based on the ipv4Network abstract data type MAY be transmitted using reduced size encoding to transmit only the network part of an IPv4 address. In ABNF-style notation, the syntax can be summed up as follows:

```
ipv4Network    =  ipv4Address unsigned8
ipv4Network    =/ *4( unsigned8 )
```

Although using an ipv4Network field instead of two separate fields for prefix and mask will not reduce the length of resulting Flow Records, it eases the work of the aggregator: With ipv4Network, the comparison of subnet addresses requires only one field lookup per Flow Record instead of two. Furthermore, using the abstract data type ipv4Network reduces the Template size by one field equalling four octets. Applications such as IPFIX Aggregation benefit from ipv4Network if network addresses are frequently exported.

5. Abstract data type portRanges

For some applications it might be useful to restrict the applicability of an Aggregation Rule to Flows with source or destination port being of a specific set of port numbers. In an Aggregation Rule, such a set of port numbers can be specified as a pattern. However, the current IPFIX Information Model does not define any data type that allows transmitting a set of port numbers, which is necessary in order to export the pattern as a Common Property of the resulting Compound Flows. Therefore, the new abstract data type portRanges for a list of port ranges is defined in

this section.

The abstract data type portRanges is a finite-length concatenation of unsigned16 value pairs, each consisting of the port range's first and last port number. Data types basing on portRanges MAY also be cast down to unsigned16 using reduced size encoding to represent a single Port. Hence, the transportSourcePort and transportDestinationPort data types, currently based on the unsigned16 abstract data type, can also be parsed as portRanges-based data types. In ABNF-style notation, the syntax can be summed up as follows:

```
portRanges     =  *(unsigned16 unsigned16)
portRanges     =/ unsigned16
```

An Information Element basing on portRanges MAY also be used as a variable-length Information Elements by prefixing it with a one-octet or three-octet length specifier as defined in [I-D.ietf-ipfix-protocol].

Table 6 shows some encoding examples with portRanges.

Port Ranges	Octets	Hexadecimal Representation
80	2	0050
1:7	4	0001 0007
80, 443	8	0050 0050 01BB 01BB
1:7, 256:1024	8	0001 0007 0100 0400
20, 80, 443	12	0014 0014 0050 0050 01BB 01BB
1:7, 80, 443	12	0001 0007 0050 0050 01BB 01BB

Table 6: PortRanges Examples

6. excludedPropertiesId Information Element

The IPFIX Information Model [I-D.ietf-ipfix-info] defines the commonPropertiesId Information Element, which can be used to link to information which several Flows have in common.

Similarly, the excludedPropertiesId shall be defined to link to a set of Common Properties which a Flow does explicitly not exhibit. An ElementId of 239 is proposed for this Information Element.

The excludedPropertiesId works like a boolean "and not" operation on the linked properties. This means that, if an excludedPropertiesId refers to a set of Common Properties which in turn specifies excluded

properties, these transitively referenced properties are to be treated as if directly referenced via a commonPropertiesId element and, hence, as being present in the Flow in question.

The excludedPropertiesId can, for example, be used when a hierarchy of Aggregation Rules with a "preceding rule" semantic, as introduced in [I-D.dressler-ipfix-aggregation], is configured in an IPFIX Aggregator.

Figure 5 illustrates the use of Common Property definitions and the linking to these definitions with Information Elements of types commonPropertiesId (CP) and excludedPropertiesId (EP). In this example, two rules are defined in the aggregator: Rule 1 matches Flows with a sourceIPv4Address of 192.0.2.1, Rule 2 matches Flows with a destinationIPv4Address of 192.0.2.2. Furthermore, Rule 1 is configured to precede Rule 2 in a hierarchy of rules, i.e. Flows that matched Rule 1 will never match Rule 2.

In order to communicate this fact to a receiver, each Aggregation Rule is transmitted as two sets of Common Properties. One set of properties (shown on the right hand side of Figure 5) directly transmits a rule's filtering criteria. The other set of properties (shown on the left hand side) refers via a commonPropertiesId to all properties that a Compound Flow exhibits, as well as via an excludedPropertiesId to all that the Compound Flow does not exhibit.

The Flow depicted at the bottom of Figure 5 thus communicates a source port of 80, a destination port of 65432, a destination IP of 192.0.2.2 and a source IP of "not 192.0.2.1". However, besides the transmission of this Flow in one Data Record, previous transmissions (and the successful reception) of four Option Templates, four Option Data Records and one Template are required to communicate this information.

```
Rule 1:
+########+------+             +######+---------------+
# CP=101 # CP=1 |<----------->#  CP=1 # SRC=192.0.2.1 |
+########+------+             +######+---------------+
                                 ^
            .--------------------'
Rule 2:     v
+########+------+------+      +######+---------------+
# CP=102 # EP=1 | CP=2 |<---->#  CP=2 # DST=192.0.2.2 |
+########+------+------+      +######+---------------+
     ^
     '-------------------.
Flow:                    v
+--------+-----------+--------+
| SPT=80 | DPT=65432 | CP=102 |
+--------+-----------+--------+
```

Figure 5: Using excludedPropertiesId to communicate a rule hierarchy

7. precedingRulePropertiesId Information Element

The only aspect in which the precedingRulePropertiesId Information Element differs from the excludedPropertiesId Information Element introduced in Section 6 is that transitive references are handled differently.

Unlike the excludedPropertiesId, the precedingRulePropertiesId does not work like a boolean "and not" operation on the linked properties. This means that, if a precedingRulePropertiesId refers to a set of Common Properties which in turn specifies excluded properties, these transitively referenced properties are to be treated as being excluded from the Flow in question, too.

Analogous to excludedPropertiesId, the precedingRulePropertiesId (PP) Information Element can be used to communicate the hierarchy of rules introduced in the example of Section 6. As illustrated in Figure 6, the amount of data transmitted is now significantly smaller, while communicating the exact same information: A source port of 80, a destination port of 65432, a destination IP of 192.0.2.2 and a source IP of "not 192.0.2.1". Besides the transmission of the Flow in one Data Record it only requires the previous transmissions (and the successful reception) of two Rich Templates.

```
          Rule 1:
          +---------------+
          | SRC=192.0.2.1 |<---.   (Rich Template 1234, CP=101)
          +---------------+    |
                               |
                               |
          Rule 2:              |
          +---------------+--------+
          | DST=192.0.2.2 | PP=101 |       (Rich Template 1235)
          +---------------+--------+


          Flow:
          +--------+-----------+
          | SPT=80 | DPT=65432 |  (Based on Rich Template 1235)
          +--------+-----------+
```

Figure 6: Using precedingRulePropertiesId to communicate a rule hierarchy

8. Security considerations

As all methods described in this document are merely variations on methods already introduced in [I-D.ietf-ipfix-protocol], the same rules regarding exchange of Flow information apply.

9. IANA Considerations

Use of the Rich Template Set requires one new IPFIX Set ID to be assigned. Use of excludedPropertiesId, precedingRulePropertiesId, as well as use of a data type basing on ipv4Network or on portRanges requires one new IPFIX Information Element identifier each to be assigned.

10. References

10.1. Normative References

[I-D.ietf-ipfix-protocol]
Claise, B., "Specification of the IPFIX Protocol for the Exchange of IP Traffic Flow Information", draft-ietf-ipfix-protocol-26 (work in progress), September 2007.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate

Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC4234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", RFC 4234, October 2005.

10.2. Informative References

[I-D.dressler-ipfix-aggregation]
Dressler, F., "IPFIX Aggregation", draft-dressler-ipfix-aggregation-03 (work in progress), June 2006.

[I-D.ietf-ipfix-info]
Quittek, J., "Information Model for IP Flow Information Export", draft-ietf-ipfix-info-15 (work in progress), February 2007.

[I-D.ietf-ipfix-reducing-redundancy]
Boschi, E., "Reducing Redundancy in IP Flow Information Export (IPFIX) and Packet Sampling (PSAMP) Reports", draft-ietf-ipfix-reducing-redundancy-04 (work in progress), May 2007.

Authors' Addresses

Christoph Sommer
University of Erlangen-Nuremberg
Department of Computer Science 7
Martensstr. 3
Erlangen 91058
Germany

Phone: +49 9131 85-27993
Email: christoph.sommer@informatik.uni-erlangen.de
URI: http://www7.informatik.uni-erlangen.de/~sommer/

Falko Dressler
University of Erlangen-Nuremberg
Department of Computer Science 7
Martensstr. 3
Erlangen 91058
Germany

Phone: +49 9131 85-27914
Email: dressler@informatik.uni-erlangen.de
URI: http://www7.informatik.uni-erlangen.de/

Gerhard Muenz
University of Tuebingen
Computer Networks and Internet
Sand 13
Tuebingen 72076
Germany

Phone: +49 7071 29-70534
Email: muenz@informatik.uni-tuebingen.de
URI: http://net.informatik.uni-tuebingen.de/

Full Copyright Statement

Intellectual Property

Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).