Network Working Group
Internet-Draft
Expires: December 26, 2006

F. Dressler
C. Sommer
University of Erlangen
G. Muenz
University of Tuebingen

June 24, 2006

IPFIX Aggregation
<draft-dressler-ipfix-aggregation-03.txt>

Status of this Memo

Copyright Notice

Abstract

IPFIX Aggregation describes a methodology for reducing the amount of measurement data exchanged between monitoring devices (IPFIX exporters) and analyzers (IPFIX collectors). Using aggregation techniques, measurement information of multiple similar flows is aggregated into one compound flow aggregate. Subsets of flows eligible for aggregation, as well as the degree of similarity, can be customized using aggregation rules.

To ensure efficient communication of both aggregated flows and the aggregation rules used, the IPFIX Protocol and IPFIX Information Model are slightly extended to allow for two new abstract data types and a new type of template set.

Table of Contents

1. Introduction

Flow measurement in high-speed large-scale networks easily produces a huge amount of data that can not be handled by a single IPFIX collector or analyzer. Also, many applications processing flow measurement data do not require detailed flow-level information but only information about flow aggregates, where the quality and level of flow aggregation is very application-specific. This document presents a flexible flow aggregation scheme that helps both, reducing the number and size of exported flow records and adapting the transmitted measurement information to the requirements of the application. These goals are achieved by discarding unneeded measurement information and merging multiple individual flows into a smaller number of compound flow aggregates before the remaining measurement data is exported to the analyzer. The following sections show how to design and implement IPFIX aggregators and introduce appropriate extensions to the IPFIX protocol.

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].

2. Terminology

Apart from the basic terms as defined in [RFC3917], [I-D.ietf-ipfix-protocol], and [I-D.ietf-ipfix-architecture], the following terms are used within this document:

Flow aggregate:
A flow aggregate contains information on one or multiple individual flows. It MAY contain the total count of all packets that belong to the same flow aggregate and were observed within a given time interval. Flow properties that were discarded during flow aggregation are no longer contained in the flow record.

Aggregation rule:
An aggregation rule defines the properties of a flow aggregate and the content of the corresponding flow record. Optionally, a set of common properties MAY be specified in order to restrict the applicability of the rule to those flows that show certain patterns.

Data Template:
A Data Template, as proposed in Section 5.3, SHOULD be used to define the structure of the flow record and to inform the analyzer about the applied aggregation rule and the common properties.

3. Architecture

Aggregation of measurement data may take place at two possible stages:

- o An internal aggregator, as depicted in Figure 1, is implemented as a process running in an IPFIX enabled device. It aggregates flow data generated by multiple metering processes and exports them as a flow aggregate. In practical implementations, metering and aggregation MAY be performed in a single step in order to reduce the number of retained state information.

```
   +---------------------------------------------+     +--------------+
   | IPFIX-enabled device       .---.   .------. |     |              |
   | .--------------------.     | A |   |      | | .-->|   Analyzer   |
   | | Metering Process 1 |-.   | g |   | E    | | |   |              |
   | `--------------------' |   | g |   | x  P | | |   +--------------+
   |                        |   | r |   | p  r |---'
   |          .             '-->| e |   | o  o | |
   |          .                 | g |-->| r  c | |
   |          .             .-->| a |   | t  e | |
   |                        |   | t |   | i  s |---.
   | .--------------------. |   | i |   | n  s | | |   +--------------+
   | | Metering Process N |-'   | o |   | g    | | |   |              |
   | `--------------------'     | n |   |      | | '-->| Concentrator |
   |                            `---'   `------' |     |              |
   +---------------------------------------------+     +--------------+
```

Figure 1: Internal Aggregation

- o An external aggregator, called concentrator in IPFIX terminology, may be used where the deployed monitoring devices cannot be modified to incorporate an internal aggregator. Furthermore, concentrators enable cascaded, multi-level aggregation of flow information. As shown in Figure 2, a concentrator receives flow records from monitoring devices and/or lower-level concentrators and exports the flow aggregate information to higher-level concentrators and/or analyzers.

```
   +------------+    +-----------------------------+    +------------+
   |            |    | Concentrator                |    |            |
   |Concentrator|-.  | .------.   .---.   .------. | .->|  Analyzer  |
   |            | |  | | C    |   | A |   | E    | | |  |            |
   +------------+ |  | | o  P |   | g |   | x  P | | |  +------------+
                  '--->| l  r |   | g |   | p  r |---'
                     | | l  o |   | r |   | o  o | |
                     | | e  c |-->| e |-->| r  c | |
                     | | c  e |   | g |   | t  e | |
                     | | t  s |   | a |   | i  s | |
                  .--->| i  s |   | t |   | n  s |---.
   +------------+ |  | | n  s |   | i |   | g  s | | |  +------------+
   |            | |  | | g    |   | o |   |      | | |  |            |
   |IPFIX device|-'  | |      |   | n |   |      | | '->|Concentrator|
   |            |    | `------'   `---'   `------' |    |            |
   +------------+    +-----------------------------+    +------------+
```

Figure 2: External Aggregation

4. Methodology

4.1 Aggregation Rules

Regarding the configuration of the aggregator, a rule-based approach is proposed. A list of user-defined aggregation rules is supplied to the aggregator. An aggregation rule consists of multiple aggregation instructions, one for each IPFIX field that is to be considered. An aggregation instruction comprises the following elements:

- The IPFIX field the aggregation instruction refers to (e.g. destinationIPv4Address). Only flows that contain the field mentioned will be considered for aggregation.
- One of the field modifiers "discard", "keep", "mask", or "aggregate" that specifies how this field is treated by the aggregator and whether it is included in the flow record or not.
- An OPTIONAL pattern for this field that restricts the aggregation to those flows that match the given value(s) (e.g. 10.10.0.0/16). Only flows that match all patterns of the rule will be aggregated.

With this definition of aggregation instructions each rule unambiguously defines the content of the flow record as well as the template to export the flow aggregate information. If a field is present in the flow record and how it is encoded depends on the field modifier. This behavior is explained in Section 4.2. Fields that do not appear in any of the aggregation instructions are not part of the flow record. The usage of patterns in order to define common properties is explained in Section 4.3.

Because the export of a flow record requires an appropriate template, a one-to-one relationship between rules and templates can be established. If a one-to-one relationship between rules and templates exits, Template Ids can serve as identifiers for the corresponding aggregation rule (see also Section 4.4).

4.2 Field Modifiers

The following field modifiers are applicable to fields that Flow Keys of incoming flow records as defined in [I-D.ietf-ipfix-architecture]. Depending on the field modifier, these fields can serve as Flow Keys of the resulting flow records. For incoming flows as well as for outgoing flow aggregates, the usage of the flowKeyIndicator is recommended for identification of the Flow Keys.

discard:
The field is not included in the flow records and is no longer a Flow Key, i.e. flow aggregates are not distinguishable with respect to this field. Incoming flow records with different values for this IPFIX field are merged.

keep:
The field remains Flow Key and is included in the flow record, i.e. flow aggregates are distinguishable with respect to this field. Incoming flow records with different values for this field are not merged, but contribute to different flow aggregates instead.

mask/n (applicable to IP addresses only):
The field is included in the flow record, but the number of significant bits is reduced to n bits. Incoming flow records with IP addresses belonging to the same subnet are merged, so flow aggregates are distinguishable with respect to resulting subnet addresses only. In accordance with the IPFIX Information Model, the resulting subnet address MAY be encoded with a IP prefix field and a IP mask field. It SHOULD, however, be encoded with a single field of the new abstract data type "ipv4Network" as proposed in Section 5.1. Independently from the encoding, the corresponding field identifying the subnet address becomes Flow Key of the flow aggregate.

In order to define a field in the flow record that does not serve as Flow Key (typically a time stamp or a count), the field modifier "aggregate" MUST be applied. Apart from being present in incoming records, there are no restrictions to the fields, i.e. they can be Flow Keys or non-Flow Keys of the original flows.

aggregate:

The field is included in the flow record but does not serve as Flow Key. The value for this field is derived from the corresponding values of the original flows. As also specified in [I-D.ietf-ipfix-info] for fields for which the value may change from packet to packet within a single flow, the value is determined by the first packet observed for the corresponding flow aggregate, unless a different semantic is explicitly specified for this Information Element. As a consequence, the value of the incoming record with the earliest start timestamp is used by default. For some Information Elements, however, a specific aggregation function is specified that has to be applied in order to get the correct value. For example, the start timestamp of the flow aggregate has to be set to the minimum of the original start timestamps, while packet and octet counts of aggregated flows are summed up. Table 1 gives an overview of such Information Elements that require a specific aggregation function. Refer to the IPFIX Information Model [I-D.ietf-ipfix-info] for a description of the mentioned fields.

Information Element	Aggregation Function
minimumPacketLength	minimum
minimumTtl	minimum
flowStartSeconds	minimum
flowStartMilliSeconds	minimum
maximumPacketLength	maximum
maximumTtl	maximum
flowEndSeconds	maximum
flowEndMilliSeconds	maximum
octetDeltaCount	sum
packetDeltaCount	sum

Table 1: Treatment of Fields Carrying Metering Information

4.3 Patterns and Common Properties

The applicability of an aggregation rule MAY be restricted to flows whose Flow Keys' values match certain patterns. Thus, patterns act as filters that enable the selection of flows and flow aggregates that are exported to the analyzer. For example, the pattern "80" can be applied to the Flow Key sourceTransportPort in order to export only (meta-)flows originated by an HTTP server. Patterns MUST NOT be used in combination with fields that are not Flow Key.

The defined patterns constitute common properties of the aggregated flows. Furthermore, the common properties are the same for all flow aggregates resulting from the corresponding aggregation rule. Common properties MAY be exported as regular IPFIX fields. However, in order to reduce redundancy and to make patterns distinguishable from other fields, they SHOULD be transmitted as fixed-value fields using the Data Template presented in Section 5.3. Additionally, encoding common properties as fixed-value fields make the applied patterns visible to the analyzer.

4.4 Rule Semantics

By default, incoming flow records are checked against all aggregation rules. If a rule matches, i.e. if the flow record comprises all required fields and matches all given patterns, the field modifiers are applied and the corresponding flow record is generated or updated. Therefore, incoming flow records that match multiple rules contribute to multiple flow aggregates.

In some cases, it is preferred that an incoming flow record that matched a certain rule is not checked against other rules in order to avoid that this flow contributes to multiple flow aggregates. Therefore, it is possible to indicate a preceding rule for each aggregation rule. If a preceding rule is given, the aggregator tries to aggregate an incoming flow according to the preceding rule. Only if the preceding rule is not applicable, e.g. because the incoming flow does not match the given pattern, the current rule is applied. Using the preceding rule relationship, rules can be organized in rule chains and rule trees where only the first matching rule is applied in every chain or branch. Consequently, each flow record is counted at most once per chain or tree. Rules that do not define a preceding rule are used to check all incoming flow records and may constitute the beginning of a rule chain or the root of a rule tree.

The Preceding Rule field in the header of the Data Template (see Section 5.3) is used to identify the preceding rule by its Template ID. If this ID is set to 0, there is no preceding rule and the rule is checked against all incoming records.

4.5 Example

Here is an example rule with four aggregation instructions:

1. Aggregate
   * discard sourceTransportPort in 80
   * keep sourceIPv4Address
   * mask/24 destinationIPv4Address in 10.10.0.0/16

   * aggregate packetDeltaCount

This rule aggregates all flows containing at least sourceTransportPort, sourceIPv4Address, destinationIPv4Address and packetDeltaCount. In addition, the destination address must be in the subnet 10.10.0.0/16 and the source port must be equal to 80. Destination addresses are merged according to subnets in 10.10.x.0/24. The resulting flow records comprise the source address, the destination subnet address, and the packet counter. The two patterns for sourceTransportPort and destinationIPv4Address are exported as fixed-value fields with the template if the Data Template specified in Section 5.3 is used. Flow that are not covered by any aggregation rule are discarded.

5. IPFIX Extensions

After having a rule's field modifiers applied, all flow records that matched a rule comprise the same fields, so for each rule exactly one template can be used. In order to efficiently transmit aggregated flows, three extensions to IPFIX are proposed:

o New abstract data type "ipv4Network"
o New abstract data type "portRanges"
o New "Data Template" set

5.1 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 new abstract data type is built of an unsigned32 for the IPv4 address and (OPTIONAL) an additional octet specifying the prefix length. The encoding of the IPv4 address corresponds to the definition of the ipv4Address in the IPFIX Information Model.

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 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.

5.2 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 flow aggregates. Therefore, the new abstract data type portRanges for a list of port ranges is defined in this section.

portRanges is a finite length string of unsigned32 values, each consisting of an unsigned16 for the first port number followed by an unsigned16 for the last port number of the port range. portRanges MAY be used as a variable-length data type as defined in [I-D.ietf-ipfix-protocol].

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, MAY be replaced portRanges-based data types.

Table 2 shows some encoding examples with portRanges.

Ports	Length	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 2: PortRanges Examples

5.3 Data Template

Section Section 4.3 described how patterns are used to restrict the applicability of an aggregation rule and define common properties of the resulting flow aggregates. In order to avoid the overhead of the repeated transmission of these common properties in all flow records resulting from a given rule, the new template type Data Template is introduced. This template type allows the exporting process to declare common properties to the analyzer. Additionally, each Data Template Record includes a Preceding Rule field that is used to inform the analyzer about the semantics of the aggregation rule sets.

The basic format of a Data Template Set is shown in Figure 3. It is the same as for a Template Set, except that the Set ID is 4. The format of individual Data Template Records, however, differs from that of the standard Template and is shown in Figure 4.

```
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               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                     Data Template Record 1                   |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                              ...                              |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                     Data Template Record N                    |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Padding (opt)                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: Data Template Set Format

The Data Template Set field definitions are as follows:

Set ID
   Type of this template set. A Set ID value of 4 is proposed for the Data Template Set.

Length
   Total length of this set in bytes.

Padding
   OPTIONAL padding to fill the set to a word boundary. It MUST consist of null-bytes and be at most seven bytes in length

```
    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                  |  Field Count                  |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  Data Count                   |  Preceding Rule               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   |                       Field 1 Specifier                       |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   |                              ...                              |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   |                       Field N Specifier                       |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   |                        Data 1 Specifier                       |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   |                              ...                              |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   |                        Data M Specifier                       |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   |                          Data 1 Value                         |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   |                              ...                              |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   |                          Data M Value                         |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4: Data Template Record Format

The Data Template Record field definitions are as follows:

Template ID
Template ID of this Data Template Record. This value is greater than 255.

Field Count
Number of regular fields that will be sent in subsequent Data Records using this Template.

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

Preceding Rule
Template ID of the preceding rule that is checked before, or 0 if all incoming records are to be checked against this rule. When a Data Template refers to a preceding rule, the exporting process SHOULD make sure that the referred Template is also exported in order to ensure that the collecting process is able to reconstruct the rule order. Refer to Section 4.4 for a description of organizing rules in chains or trees.

Field N Specifier
Information Element identifier, Field length and an Enterprise Number (if needed) 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 3 illustrates the relationship between field modifiers and common properties (defined as patterns) on the one hand, and the resulting regular and fixed-value fields in the Data 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 Data 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 Data 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 3: Relation between field modifiers, flow records, and Data Templates

5.4 Example

In this example we assume the concentrator was given the following aggregation rule:

1. Aggregate
   - discard sourceIPv4Address in 10.0.0.0/23
   - keep destinatonTransportPort
   - aggregate packetDeltaCount

We further assume the concentrator receives the following flow records:

Source IP	Source Port	Destination IP	Destination Port	Packets
10.0.0.1	64235	10.0.0.10	80	10
10.0.1.2	64236	10.0.0.11	110	10
10.0.0.3	64237	10.0.0.12	80	10
10.0.2.4	64238	10.0.0.13	80	10
10.0.2.5	64239	10.0.0.14	80	10

Table 4: Incoming Flows

Based on the aggregation rule stated above the concentrator would now first send a Data Template Set with the Data Template Record corresponding to the given rule:

Field	Value
Template ID	10001
Field Count	2
Data Count	2
Preceding Rule	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	10.0.0.0
Data 2 Value	23

Table 5: Data Template used

In case that the abstract data type ipv4Network was used for a new data type Source IP Network, it would look like this:

Field	Value
Template ID	10001
Field Count	2
Data Count	2
Preceding Rule	0
Field 1 Type	Destination Port
Field 2 Type	Packets
Data 1 Type	Source IP Network
Data 1 Value	10.0.0.0/23

Table 6: Data Template used

Secondly, a Data Set of this Data Template is exported containing the flow aggregates resulting from the given rule. Note that the flows' common property, a source IP address in 10.0.0.0/23, was already transmitted in the template. The exported flow records contain the aggregated packet counts and the destination port, which is the only discriminating Flow Key property.

Destination Port	Packets
80	20
110	10

Table 7: Aggregated Flows

6. Application Examples

6.1 Charging

Charging applications require separate flow accounting for individual end systems. However, detailed information about all individual flows sent or received by the end system is not required. The required level of flow aggregation can be achieved with an aggregation rules that discards all Flow Key properties except the IP address of the involved end systems.

The example ruleset can be used for charging end systems in the subnet 10.10.0.0/16:

1. Aggregate
    * keep destinationIPv4Address in 10.10.0.0/16
    * aggregate packetDeltaCount
2. Aggregate
    * keep sourceIPv4Address in 10.10.0.0/16
    * aggregate packetDeltaCount

flow records resulting from the first rule contain packet counts of the inbound traffic separated by host IP addresses. The second rule produces the corresponding records for the outbound traffic. Protocol and port information is omitted.

6.2 Intrusion Detection

If flow accounting is employed for intrusion detection, e.g. in order to detect denial-of-service attacks, information about the transport layer protocol and attacked service, i.e. the destination port, is mostly required. On the other hand, the analysis is typically based on flow aggregates exchanged between subnets since processing individual flows would require to much processing power. Detailed information about the flows between individual end systems is only required if an already detected attack should be analyzed in more detail.

An example ruleset might consist of the following instructions:

1. Aggregate
   - mask/24 destinationIPv4Address in 10.10.0.0/16
   - mask/24 sourceIPv4Address
   - keep protocolIdentifier
   - keep destinationTransportPort
   - aggregate packetDeltaCount

flow records are created for all packets directed to /24 subnets in the protected network 10.10.0.0/16. The destination port and the protocol are preserved whereas the source port is discarded.

7. 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.

8. References

8.1 Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", March 1997.

8.2 Informative References

[I-D.ietf-ipfix-architecture]
Sadasivan, G., Brownlee, N., Claise, B., and J. Quittek, "Architecture for IP Flow Information Export", draft-ietf-ipfix-architecture-11 (work in progress), June 2006.

[I-D.ietf-ipfix-protocol]
Claise, B., "IPFIX Protocol Specifications", draft-ietf-ipfix-protocol-22 (work in progress), June 2006.

[I-D.ietf-ipfix-info]
Quittek, J., Bryant, S., Claise, B., and J. Meyer, "Information Model for IP Flow Information Export", draft-ietf-ipfix-info-12 (work in progress), June 2006.

[RFC3917] Quittek, J., Zseby, T., Claise, B., and S. Zander, "Requirements for IP Flow Information Export", RFC 3917, October 2004.

Authors' Addresses

Falko Dressler
University of Erlangen
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/

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

Email: sichsomm@stud.informatik.uni-erlangen.de

Gerhard Muenz
University of Tuebingen
Computer Networks and Internet
Auf der Morgenstelle 10C
Tuebingen 72076
Germany

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

Intellectual Property Statement

Disclaimer of Validity

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Copyright Statement

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.