

IPFIX Working Group
Internet-Draft
Intended Status: Standards Track
Expires: August 20, 2011

B. Claise
Cisco Systems, Inc.
A. Kobayashi
NTT PF Lab.
B. Trammell
ETH Zurich
February 14, 2011

Specification of the Protocol for IPFIX Mediations
draft-claise-ipfix-mediation-protocol-03

Abstract

This document specifies the IP Flow Information Export (IPFIX) protocol specific to the Mediation.

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at <http://www.ietf.org/ietf/lid-abstracts.txt>

The list of Internet-Draft Shadow Directories can be accessed at <http://www.ietf.org/shadow.html>

This Internet-Draft will expire on April, 2011.

Copyright Notice

Copyright (c) 2011 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents

Internet-Draft <Protocol for IPFIX Mediations> February 2011
carefully, as they describe your rights and restrictions with
respect to this document. Code Components extracted from this
document must include Simplified BSD License text as described
in Section 4.e of the Trust Legal Provisions and are provided
without warranty as described in the Simplified BSD License.

Conventions used in this document

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 RFC 2119 [RFC2119].

Table of Contents

1. Introduction.....	3
1.1. IPFIX Documents Overview.....	4
1.2. IPFIX Mediator Documents Overview.....	4
1.3. Relationship with IPFIX and PSAMP.....	5
2. Terminology.....	6
3. Specifications.....	9
3.1. Encoding of IPFIX Message Header.....	10
3.2. Template Management.....	11
3.2.1. Template Management Without Template Records Change	
3.2.2. Template Management With New Template Records....	14
3.3. Time Management.....	17
3.4. Observation Point Management.....	19
3.4.1. Observation Domain Management.....	20
3.5. Specific Reporting Requirements.....	21
3.5.1. The Flow Keys Options Template.....	22
3.5.2. IPFIX Protocol Options Template.....	22
3.5.3. IPFIX Mediator Options Template.....	23
3.6. The Collecting Process's Side.....	23
3.7. Configuration Management.....	24
4. New Information Elements.....	24
4.1. - originalExporterIPv4Address.....	24
4.2. originalExporterIPv6Address.....	24
4.3. originalObservationDomainId.....	25
5. Security Considerations.....	25
6. IANA Considerations.....	26
6.1. originalExporterIPv4Address.....	26
6.2. originalExporterIPv6Address.....	26
6.3. originalObservationDomainId.....	27
7. References.....	27

Internet-Draft	<Protocol for IPFIX Mediations>	February 2011
7.1. Normative References.....		27
7.2. Informative References.....		28
8. Author's Addresses.....		29
9. Appendix A. Additions to XML Specification of IPFIX Information Elements.....		30

1. Introduction

The IPFIX architectural components in [RFC5470] consist of IPFIX Devices and IPFIX Collectors communicating using the IPFIX protocol [RFC5101], which specifies how to export IP Flow information. This protocol is designed to export information about IP traffic Flows and related measurement data, where a Flow is defined by a set of key attributes (e.g. source and destination IP address, source and destination port, etc.).

However, thanks to its Template mechanism, the IPFIX protocol can export any type of information, as long as the relevant Information Element is specified in the IPFIX Information Model [RFC5102], registered with IANA, or specified as an enterprise-specific Information Element. The specifications in the IPFIX protocol [RFC5101] have not been defined in the context of an IPFIX Mediator receiving, aggregating, correlating, anonymizing, etc... Flow Records from the one or multiple Exporters. Indeed, the IPFIX protocol must be adapted for Intermediate Processes, as defined in the IPFIX Mediation Reference Model as specified in the Figure A of [IPFIX-MED-FMWK], which is based on the IPFIX Mediation Problem Statement [RFC5982].

This document specifies the IP Flow Information Export (IPFIX) protocol in the context of the implementation and deployment of IPFIX Mediators. The use of the IPFIX protocol within a Mediator -- a device which contains both as an Exporting Process and a Collecting Process -- has an impact on the technical details of the usage of the protocol. An overview of the technical problem is covered in section 6 of the [RFC5982]: loss of original exporter information, loss of base time information, transport sessions management, loss of Options Template Information, Template Id management, considerations for network topology, and IPFIX Mediation interpretation, and considerations for aggregation.

The specifications in this document are based on the IPFIX protocol specifications but adapted according to the IPFIX Mediation Framework [IPFIX-MED-FMWK].

1.1. IPFIX Documents Overview

The IPFIX Protocol [RFC5101] provides network administrators with access to IP Flow information.

The architecture for the export of measured IP Flow information out of an IPFIX Exporting Process to a Collecting Process is defined in the IPFIX Architecture [RFC5470], per the requirements defined in RFC 3917 [RFC3917].

The IPFIX Architecture [RFC5470] specifies how IPFIX Data Records and Templates are carried via a congestion-aware transport protocol from IPFIX Exporting Processes to IPFIX Collecting Processes.

IPFIX has a formal description of IPFIX Information Elements, their name, type and additional semantic information, as specified in the IPFIX Information Model [RFC5102].

The IPFIX Applicability Statement [RFC5472] describes what type of applications can use the IPFIX protocol and how they can use the information provided. It furthermore shows how the IPFIX framework relates to other architectures and frameworks.

"IPFIX Mediation: Problem Statement" [RFC5982], describing the IPFIX Mediation applicability examples, along with some problems that network administrators have been facing, is the basis for the "IPFIX Mediation: Framework" [IPFIX-MED-FMWK]. This framework details the IPFIX Mediation reference model and the components of an IPFIX Mediator.

1.2. IPFIX Mediator Documents Overview

The "IPFIX Mediation: Problem Statement" [RFC5982] provides an overview of the applicability of Mediators, and defines requirements for Mediators in general terms. This document is of use largely to define the problems to be solved through the deployment of IPFIX Mediators, and to provide scope to the role of Mediators within an IPFIX collection infrastructure.

Internet-Draft <Protocol for IPFIX Mediations> February 2011
The "IPFIX Mediation: Framework" [IPFIX-MED-FMWK] provides more architectural details of the arrangement of Intermediate Processes within a Mediator.

The details of specific Intermediate Processes, when these have additional export specifications (e.g., metadata about the intermediate processing conveyed through IPFIX Options Templates), are each treated in their own document (e.g., the "IP Flow Anonymisation Support" [IPFIX-MED-ANON]). Documents specifying the operations of specific Intermediate Processes cover the operation of these Processes within the Mediator framework, and complying to the specifications given in this document; they may additionally specify the operation of the process independently, outside the context of a Mediator, when this is appropriate. As of today, these documents are:

1. "IP Flow Anonymisation Support", [IPFIX-MED-ANON], which describes anonymisation techniques for IP flow data and the export of anonymised data using the IPFIX protocol.
2. "Flow Selection Techniques" [IPFIX-MED-FLOWSEL], which described the process of selecting a subset of flows from all flows observed at an observation point, along with the motivations, and some specific flow selection techniques.
3. "Exporting Aggregated Flow Data using the IP Flow Information Export" [IPFIX-MED-AGGR] which describes Aggregated Flow export within the framework of IPFIX Mediators and defines an interoperable, implementation-independent method for Aggregated Flow export.

1.3. Relationship with IPFIX and PSAMP

The specification in this document applies to the IPFIX protocol specifications [RFC5101]. All specifications from [RFC5101] apply unless specified otherwise in this document.

As the Packet Sampling (PSAMP) protocol specifications [RFC5476] are based on the IPFIX protocol specifications, the specifications in this document are also valid for the PSAMP protocol. Therefore, the method specified by this document also applies to PSAMP.

The IPFIX-specific terms, such as Observation Domain, Flow, Flow Key, Metering Process, Exporting Process, Exporter, IPFIX Device, Collecting Process, Collector, Template, IPFIX Message, Message Header, Template Record, Data Record, Options Template Record, Set, Data Set, Information Element, and Transport Session, used in this document are defined in [RFC5101]. The PSAMP-specific terms used in this document, such as Filtering and Sampling are defined in [RFC5476].

The IPFIX Mediation terms related to the aggregation, such as the Interval, Aggregated Flow, and Aggregated Function are defined in [IPFIX-MED-AGGR].

The IPFIX Mediation-specific terminology used in this document is defined in "IPFIX Mediation: Problem Statement" [RFC5982], and reuse in "IPFIX Mediation: Framework" [IPFIX-MED-FMWK]. However, since those two documents are an informational RFC, the definitions have been reproduced here along with additional definitions.

Similarly, since the [IPFIX-MED-ANON] is an experimental RFC, the Anonymisation Record, Anonymised Data Record, and Intermediate Anonymisation Process terms, specified in [IPFIX-MED-ANON], are also reproduced here.

In this document, as in [RFC5101], [RFC5476], [IPFIX-MED-AGGR], and [IPFIX-MED-ANON], the first letter of each IPFIX-specific and PSAMP-specific term is capitalized along with the IPFIX Mediation-specific term defined here. In this document, we call "record stream" a stream of records carrying flow- or packet-based information. The records may be encoded as IPFIX Data Records in any other format.

Transport Session Information

The Transport Session is specified in [RFC5101]. In SCTP, the Transport Session Information is the SCTP association. In TCP and UDP, the Transport Session Information corresponds to a 5-tuple {Exporter IP address, Collector IP address, Exporter transport port, Collector transport port, transport protocol}.

Original Exporter

An Original Exporter is an IPFIX Device that hosts the Observation Points where the metered IP packets are observed.

An Observation Point of the Original Exporter(s). In the case of the Intermediate Aggregation Process on an IPFIX Mediator, the Original Observation Point can be composed of a (set of) specific exporter(s), a (set of) specific interface(s) on an Exporter, a (set of) line card(s) on an Exporter, or any combinations of these.

IPFIX Mediation

IPFIX Mediation is the manipulation and conversion of a record stream for subsequent export using the IPFIX protocol.

The following terms are used in this document to describe the architectural entities used by IPFIX Mediation.

Intermediate Process

An Intermediate Process takes a record stream as its input from Collecting Processes, Metering Processes, IPFIX File Readers, other Intermediate Processes, or other record sources; performs some transformations on this stream, based upon the content of each record, states maintained across multiple records, or other data sources; and passes the transformed record stream as its output to Exporting Processes, IPFIX File Writers, or other Intermediate Processes, in order to perform IPFIX Mediation. Typically, an Intermediate Process is hosted by an IPFIX Mediator. Alternatively, an Intermediate Process may be hosted by an Original Exporter.

Specific Intermediate Processes are described below. However, this is not an exhaustive list.

Intermediate Conversion Process

An Intermediate Conversion Process is an Intermediate Process that transforms non-IPFIX into IPFIX, or manages the relation among Templates and states of incoming/outgoing Transport Sessions (or equivalent for non IPFIX protocols) in the case of transport protocol conversion (e.g., from UDP to SCTP).

Intermediate Aggregation Process

An Intermediate Aggregation Process is an Intermediate Process that aggregates records based upon a set of Flow Keys or functions applied to fields from the record (e.g., binning and subnet aggregation).

Intermediate Correlation Process

An Intermediate Correlation Process is an Intermediate Process that adds information to records, noting correlations among them, or generates new records with correlated data from multiple records (e.g., the production of bidirectional flow records from unidirectional flow records).

Intermediate Selection Process

An Intermediate Selection Process is an Intermediate Process that selects records from a sequence based upon criteria-evaluated record values and passes only those records that match the criteria (e.g., Filtering only records from a given network to a given Collector).

Intermediate Anonymization Process

An Intermediate Anonymization Process is an Intermediate Process that transforms records in order to anonymize them, to protect the identity of the entities described by the records (e.g., by applying prefix-preserving pseudonymization of IP addresses).

IPFIX Mediator

An IPFIX Mediator is an IPFIX Device that provides IPFIX Mediation by receiving a record stream from some data sources, hosting one or more Intermediate Processes to transform that stream, and exporting the transformed record stream into IPFIX Messages via an Exporting Process. In the common case, an IPFIX Mediator receives a record stream from a Collecting Process, but it could also receive a record stream from data sources not encoded using IPFIX, e.g., in the case of conversion from the NetFlow V9 protocol [RFC3954] to IPFIX protocol.

Template Mapping

A mapping from Template Records and/or Options Template Records received by a Mediator to Template Records and/or Options Template Records sent by that IPFIX Mediator. Each

Internet-Draft <Protocol for IPFIX Mediations> February 2011
entry in a Template Mapping is scoped by incoming or outgoing
Transport Session and Observation Domain, as with Templates
and Options Templates in the IPFIX Protocol.

Anonymisation Record

A record, defined by the Anonymisation Options Template in
section Section 6.1, that defines the properties of the
anonymisation applied to a single Information Element within a
single Template or Options Template.

Anonymised Data Record

A Data Record within a Data Set containing at least one
Information Element with anonymised values. The Information
Element(s) within the Template or Options Template describing
this Data Record SHOULD have a corresponding Anonymisation
Record.

Intermediate Anonymisation Process

An intermediate process which takes Data Records and and
transforms them into Anonymised Data Records.

3. Specifications

This section describes the IPFIX specifications for Mediation:
more specifically, specifications for generic Intermediate
Processes. Possible specific Intermediate Processes are:
Intermediate Conversion Process, Intermediate Aggregation
Process, Intermediate Correlation Process, Intermediate
Selection Process, Intermediate Anonymization Process.

For a specific Intermediate Process, the specifications in the
following reference MUST be followed, on the top of the
specifications in this document:

- For the Intermediate Aggregation Process, the specifications
in [IPFIX-MED-AGGR] MUST be followed.
- For the Intermediate Selection Process, the specifications in
[IPFIX-MED-FLOWSEL] MUST be followed.
- For the Intermediate Anonymization Process, the specifications
in [IPFIX-MED-ANON] should be considered as guidelines as
[IPFIX-MED-ANON] is an experimental RFC.

Note that no specific document deals with the Intermediate
Conversion Process at the time of this publication.

Internet-Draft <Protocol for IPFIX Mediations> February 2011
 These new specifications, which are more specific compared to
 [RFC5101], are described with the key words described in
 [RFC2119].

3.1. Encoding of IPFIX Message Header

The format of the IPFIX Message Header is shown in Figure A.
 Note that the format is similar to the IPFIX Message in
 [RFC5101], but some field definitions (for the example, the
 Export Time) have been updated in the context of the IPFIX
 Mediator.

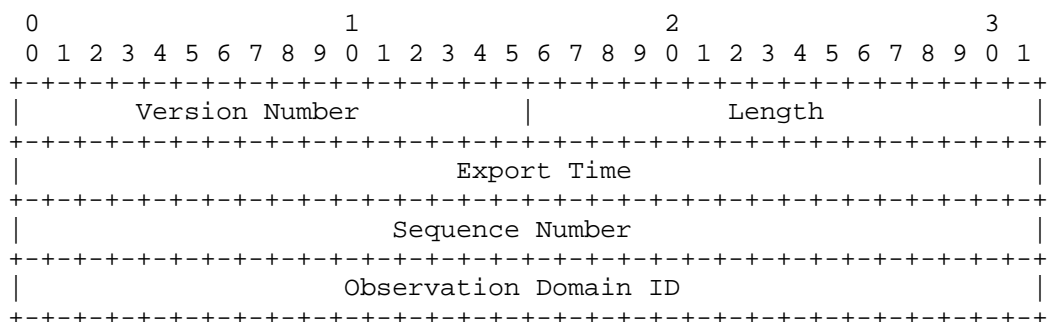


Figure A: IPFIX Message Header format

Message Header Field Descriptions

Version

Version of Flow Record format exported in this message.
 The value of this field is 0x000a for the current
 version, incrementing by one the version used in the
 NetFlow services export version 9 [RFC3954].

Length

Total length of the IPFIX Message, measured in octets,
 including Message Header and Set(s).

Export Time

Sequence Number

Incremental sequence counter modulo 2^{32} of all IPFIX Data Records sent on this PR-SCTP stream from the current Observation Domain by the Exporting Process. Check the specific meaning of this field in the sub-sections of section 10 when UDP or TCP is selected as the transport protocol. This value SHOULD be used by the Collecting Process to identify whether any IPFIX Data Records have been missed. Template and Options Template Records do not increase the Sequence Number.

Observation Domain ID

A 32-bit identifier of the Observation Domain that is locally unique to the Exporting Process. The Exporting Process uses the Observation Domain ID to uniquely identify to the Collecting Process the Observation Domain that metered the Flows. It is RECOMMENDED that this identifier is also unique per IPFIX Device. Collecting Processes SHOULD use the Transport Session and the Observation Domain ID field to separate different export streams originating from the same Exporting Process. The Observation Domain ID SHOULD be 0 when no specific Observation Domain ID is relevant for the entire IPFIX Message. For example, when exporting the Exporting Process Statistics, or in case of hierarchy of Collector when aggregated Data Records are exported.
Note: the Observation Domain Management is discussed in section 3.4.1.

3.2. Template Management

3.2.1. Template Management Without Template Records Change

The first case is a situation where the IPFIX Mediator doesn't modify the (Options) Template Record(s) content. A typical example is an Intermediate Selection Process acting as distributor, which collects Flow Records from one or multiple Exporters, and based on the Information Elements content, redirects the Flow Records to the appropriate Collector. This example is a typical case of a single network operation center

Internet-Draft <Protocol for IPFIX Mediations> February 2011
managing multiple universities: an unique IPFIX Collector
collects all Flow Records for the common infrastructure, but
might be re-exporting specific university Flow Records to the
responsible system administrator.

As specified in [RFC5101], the Template IDs are unique per
Exporter, per Transport Session, and per Observation Domain. As
there is no guarantee that, for similar Template Records, the
Template IDs received on the incoming Transport Session and
exported to the outgoing Transport Session would be same, the
IPFIX Mediator MUST maintain a Template Mapping composed of
similar received and exported (Options) Template Records:

- for each received (Options) Template Record: Template Record
Flow Keys and non Flow Keys, Template ID, Observation Domain
Id, and Transport Session
- for each exported (Options) Template Record: Template Record
Flow Keys and non Flow Keys, Template ID, Collector,
Observation Domain Id, and Transport Session

If an IPFIX Mediator receives an IPFIX Withdrawal Message for a
(Options) Template Record that is not used anymore in any
outgoing Transport Sessions, the IPFIX Mediator SHOULD export
the appropriate IPFIX Withdrawal Message(s) on the outgoing
Transport Session, and remove the corresponding entry in the
Template Mapping.

If a (Options) Template Record is not used anymore in an
outgoing Transport Session, it MUST be withdrawn with an IPFIX
Template Withdrawal Message on that specific outgoing Transport
Session, and its entry MUST be removed from the Template
Mapping.

If an incoming or outgoing Transport Session is gracefully
shutdown or reset, the (Options) Template Records corresponding
to that Transport Session MUST be removed from the Template
Mapping.

Figure B displays an example of an Intermediate Selection
Process, re-distributing Data Records to Collectors on the basis
of the customer networks, i.e. the Route Distinguisher (RD). In
this example, the Template Record received from the Exporter#1
is reused towards the Collector#1, Collector#2, and Collector#3.

```
Templ. .-----.  
ID 256 |          |
```

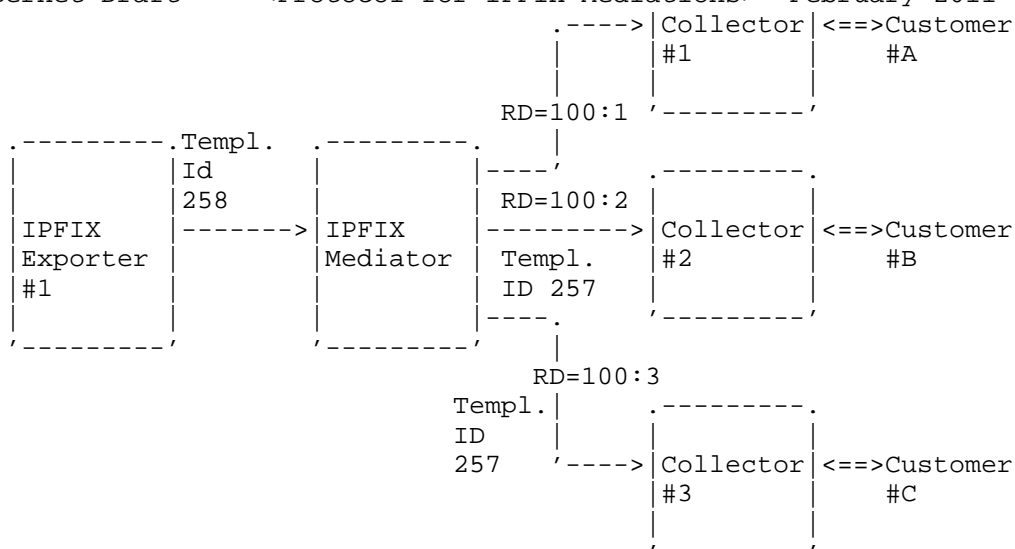


Figure B: Intermediate Aggregation Process Example

Template Entry A:

Incoming Transport Session (from Exporter#1):

Source IP: <Exporter#1 export IP address>
 Destination IP: <IPFIX Mediator IP address>
 Source Port: <source port>
 Destination Port: 4739 (IPFIX)
 Observation Domain Id: <Observation Domain ID>
 Template Id: 258
 Flow Keys: <series of Flow Keys>
 Non Flow Keys: <series of non Flow Keys>

Template Entry B:

Outgoing Transport Session (to Collector#1):

Source IP: <IPFIX Mediator IP address>
 Destination IP: <IPFIX Collector#1 IP address>
 Source Port: <source port>
 Destination Port: 4739 (IPFIX)
 Observation Domain Id: <Observation Domain ID>
 Template Id: 256
 Flow Keys: <series of Flow Keys>
 Non Flow Keys: <series of non Flow Keys>

Template Entry C:

Outgoing Transport Session (to Collector#2):

Internet-Draft <Protocol for IPFIX Mediations> February 2011

 Source IP: <IPFIX Mediator IP address>

 Destination IP: <IPFIX Collector#2 IP address>

 Source Port: <source port>

 Destination Port: 4739 (IPFIX)

 Observation Domain Id: <Observation Domain ID>

 Template Id: 257

 Flow Keys: <series of Flow Keys>

 Non Flow Keys: <series of non Flow Keys>

Template Entry D:

 Outgoing Transport Session (to Collector#3):

 Source IP: <IPFIX Mediator IP address>

 Destination IP: <IPFIX Collector#3 IP address>

 Source Port: <source port>

 Destination Port: 4739 (IPFIX)

 Observation Domain Id: <Observation Domain ID>

 Template Id: 257

 Flow Keys: <series of Flow Keys>

 Non Flow Keys: <series of non Flow Keys>

The Template Mapping corresponding to the figure B can be displayed as:

 Template Entry A <----> Template Entry B

 Template Entry A <----> Template Entry C

 Template Entry A <----> Template Entry D

3.2.2. Template Management With New Template Records

The second case is a situation where the IPFIX Mediator generates new (Options) Template Records compared to the received ones.

In such a situation, the IPFIX Mediator doesn't need to maintain a Template Mapping, as it generates its own series of (Options) Template Records. However, the following special case might still require a Template Mapping, i.e. a situation where the IPFIX Mediator, typically containing an Intermediate Conversion Process, Intermediate Aggregation Process [IPFIX-MED-AGGR], or Intermediate Anonymization Process in case of black-marker anonymisation [IPFIX-MED-ANON], generates new (Options) Template Records based on what it receives from the Exporter(s), and based on the Intermediate Process function. In such a case, it's interesting to keep the correlation between the received (Options) Template Records and exported Derived Options) Template Records in the Template Mapping.

Therefore, the IPFIX Mediator MAY maintain a Template Mapping composed of received (Options) Template Records and exported derived Options) Template Records:

- for each received (Options) Template Record: Template Record Flow Keys and non Flow Keys, Template ID, Observation Domain, and Transport Session
- for each exported derived Options) Template Record: Template Record Flow Keys and non Flow Keys, Template ID, Collector, Observation Domain, and Transport Session

If an IPFIX Mediator receives an IPFIX Withdrawal Message for a (Options) Template Record that is not used anymore as the basis of an inferred (Options) Template Records, the IPFIX Mediator SHOULD export the appropriate IPFIX Withdrawal Message(s) for the inferred (Options) Template Record on the outgoing Transport Session, and remove the corresponding entry in the Template Mapping.

The following two examples illustrate this.

First, consider an IPFIX Mediator hosting an Intermediate Aggregation Process that generates time-series traffic octet counts per source IP address (as in the example in section 8.1 of [IPFIX-MED-AGGR]). Here, the Intermediate Process accepts Flow Records fitting any Template, discards all Information Elements other than the sourceIPv[46]Address and octetDeltaCount, aggregates these across all original Exporters in a given regular time interval, and exports Flow Records according to a Template Record containing flowStartTimeMilliseconds, flowEndTimeMilliseconds, sourceIPv[46]Address, and octetDeltaCount.

In this case, no Template Mapping is necessary. New Templates and Template Withdrawals in the Transport Sessions from the Original Exporters are handled as they would be at any Collecting Process. Records according to Templates which do not contain at least a timestamp, sourceIPv[46]Address, and octetDeltaCount IE are simply discarded by the Collector.

Next, consider a more generic case of this Intermediate Aggregation Process, which creates time-series aggregates across all Original Exporters, imposing a time interval but keeping a subset of the incoming Flow Key received from the Original Exporter. In this case, a Template Mapping is necessary, as there is a relationship between incoming and outgoing Templates.

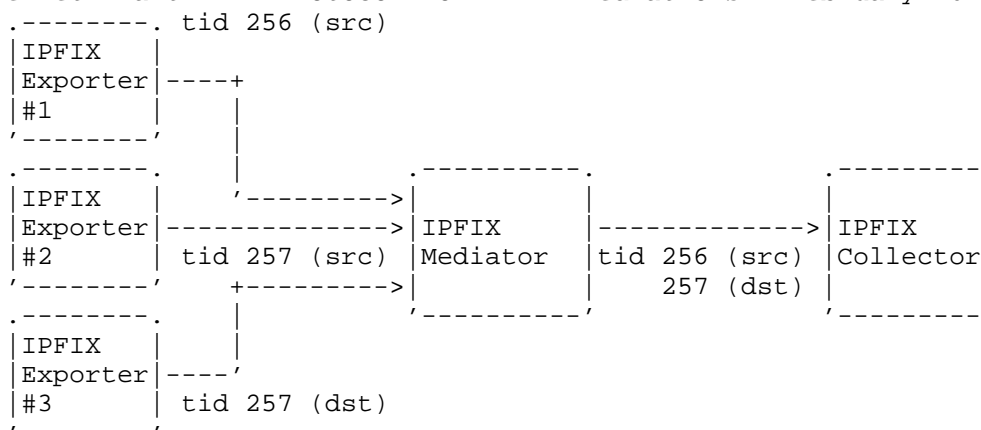


Figure C: Intermediate Aggregation Process Example

In Figure C, above, the Mediator accepts a Template Record containing only the sourceIPv4Address as the Flow Key from Exporters 1 and 2, and a Template Record containing only the destinationIPv4Address as the Flow Key from exporter 3. It exports time-series source aggregates as Template ID 256, and time-series destination aggregates as Template ID 257. The Template Entries in this case are as follows:

Template Entry A:

```

Incoming Transport Session (from Exporter#1):
    Source IP: <Exporter#1 export IP address>
    Destination IP: <IPFIX Mediator IP address>
    Source Port: <source port>
    Destination Port: 4739 (IPFIX)
Observation Domain Id: <Observation Domain ID>
Template Id: 256
    Flow Keys: sourceIPv4Address
    Non Flow Keys: octetDeltaCount, [others]
  
```

Template Entry B:

```

Incoming Transport Session (from Exporter#2):
    Source IP: <Exporter#2 export IP address>
    Destination IP: <IPFIX Mediator IP address>
    Source Port: <source port>
    Destination Port: 4739 (IPFIX)
Observation Domain Id: <Observation Domain ID>
Template Id: 257
    Flow Keys: sourceIPv4Address
    Non Flow Keys: octetDeltaCount, [others]
  
```

Template Entry C:

 Incoming Transport Session (from Exporter#3):

 Source IP: <Exporter#3 export IP address>

 Destination IP: <IPFIX Mediator IP address>

 Source Port: <source port>

 Destination Port: 4739 (IPFIX)

 Observation Domain Id: <Observation Domain ID>

 Template Id: 257

 Flow Keys: destinationIPv4Address

 Non Flow Keys: octetDeltaCount, [others]

Template Entry D:

 Outgoing Transport Session (to IPFIX Collector):

 Source IP: <IPFIX Mediator export IP address>

 Destination IP: <IPFIX Collector IP address>

 Source Port: <source port>

 Destination Port: 4739 (IPFIX)

 Observation Domain Id: <Observation Domain ID>

 Template Id: 256

 Flow Keys: sourceIPv4Address

 Non Flow Keys: octetDeltaCount

Template Entry E:

 Outgoing Transport Session (to IPFIX Collector):

 Source IP: <IPFIX Mediator export IP address>

 Destination IP: <IPFIX Collector IP address>

 Source Port: <source port>

 Destination Port: 4739 (IPFIX)

 Observation Domain Id: <Observation Domain ID>

 Template Id: 257

 Flow Keys: destinationIPv4Address

 Non Flow Keys: octetDeltaCount

The Template Mapping corresponding to the figure C can be displayed as:

 Template Entry A <----> Template Entry D

 Template Entry B <----> Template Entry D

 Template Entry C <----> Template Entry E

3.3. Time Management

The IPFIX Message Header "Export Time" field is the time in seconds since 0000 UTC Jan 1, 1970, at which the IPFIX Message Header leaves the IPFIX Mediator. However, in the specific case

Internet-Draft <Protocol for IPFIX Mediations> February 2011
of an IPFIX Mediator containing an Intermediate Conversion
Process, the IPFIX Mediator MAY keep the export time received
from the incoming Transport Session.

It is RECOMMENDED that Mediators handle time using absolute
timestamps (e.g. flowStartSeconds, flowStartMilliseconds,
flowStartNanoseconds), which are specified relative to the UNIX
epoch (00:00 UTC 1 Jan 1970), where possible, rather than
relative timestamps (e.g. flowStartSysUpTime,
flowStartDeltaMicroseconds), which are specified relative to
protocol structures such as system initialization or message
export time.

The latter are difficult to manage for two reasons. First, they
require constant translation, as the system initialization time
of an intermediate system and the export time of an intermediate
message will change across mediation operations. Further,
relative timestamps introduce range problems. For example, when
using the flowStartDeltaMicroseconds and
flowEndDeltaMicroseconds Information Elements [RFC5102], the
Data Record must be exported within a maximum of 71 minutes
after its creation. Otherwise, the 32-bit counter would not be
sufficient to contain the flow start time offset. Those time
constraints might be incompatible with some of the Intermediate
Processes: Intermediate Aggregation Process (temporal) and
Intermediate Correlation Process, for example.

When an Intermediate Aggregation Process aggregates information
from different Flow Records, the typical reporting times SHOULD
BE the minimum of the start times and the maximum of the end
times. However, if the Flow Records do not overlap, i.e. if
there is a time gap between the times in the Flow Records, then
the report may be inaccurate. The IPFIX Mediator is only
reporting what it knows, on the basis of the information made
available to it - and there may not have been any data to
observe during the gap. Then again, if there is an overlap in
timestamps, there's the potential of double-accounting:
different Observation Points may have observed the same traffic
simultaneously. Therefore, as there is not a single rule that
fits all different situations, the precise rules of applying the
Flow Record timestamps in IPFIX Mediators is out of the scope of
this document. However, some more specifications related to the
specific case of aggregation in space and time are specified in
[IPFIX-MED-AGGR], and MUST be followed.

Depending on the use case, top Collectors may need to receive the Original Observation Point(s), otherwise it may wrongly conclude that the IPFIX Device exporting the Flow Records to him, i.e. the IPFIX Mediator, directly observed the packets that generated the Flow Records. Two new Information Element are introduced to solve this use case: `originalExporterIPv4Address` and `originalExporterIPv6Address`.

In the IPFIX Mediator, the Observation Point(s) may be represented by:

- A single Original Exporter (represented by the `originalExporterIPv4Address` or `originalExporterIPv6Address` Information Elements)
- A list of Original Exporter (represented by the `originalExporterIPv4Address` or `originalExporterIPv6Address` Information Elements_)
- A list of Original Exporter (represented by the `originalExporterIPv4Address` or `originalExporterIPv6Address` Information Elements), along with the associated interface (represented by the `ingressInterface` and/or `egressInterface`)
- A list of Original Exporter (represented by the `originalExporterIPv4Address` or `originalExporterIPv6Address` Information Elements), along with the associated line card id (represented by the `lineCardId`)
- Any combination or list of Information Elements representing Observation Points.

Some Information Elements characterizing the Observation Point may be added. For example, the `flowDirection` Information Element specifies the direction of the observation, and, as such, characterizes the Observation Point.

Any combination of the above examples is possible. For example, in case of an Intermediate Aggregation Process, an Original Observation Point can be composed of:

```
exporterIPv4Address 192.0.2.1
exporterIPv4Address 192.0.2.2,
    interface ethernet 0, direction ingress
    interface ethernet 1, direction ingress
    interface serial 1, direction egress
    interface serial 2, direction egress
exporterIPv4Address 192.0.2.3,
    lineCardId 1, direction ingress
```

If the Original Observation Point is composed of a list, then the IPFIX Structured Data [IPFIX-STRUCT] MUST be used to export it from the IPFIX Mediator.

The most generic way to export the Original Observation Point is to use a subTemplateMultiList, with the semantic "exactlyOneOf". Taking back the previous example, the following encoding can be used:

```
Template Record 257: exporterIPv4Address
Template Record 258: exporterIPv4Address, basicList of
                    ingressInterface, flowDirecdtion
Template Record 259: exporterIPv4Address, lineCardId,
                    flowDirection
```

The Original Observation Point is modeled with the Data Records corresponding to either Template Record 1, Template Record 2, or Template Record 3 but not more than one of these ("exactlyOneOf" semantic). This implies that the Flow was observed at exactly one of the Observation Points reported.

When an IPFIX Mediator receives Flow Records containing the Original Observation Point Information Element, i.e. originalExporterIPv6Address or originalExporterIPv4Address, the IPFIX Mediator SHOULD NOT modify its value(s) when composing new Flow Records in the general case. Known exceptions include anonymization per [IPFIX-MED-ANON] section 7.2.4 and an Intermediate Correlation Process rewriting addresses across NAT.

In other words, the Original Observation Point should not be replaced the IPFIX Mediator Observation Point. The daisy chain of (Exporter, Observation Point) representing the path the Flow Records took from the Exporter to the top Collector, via the IPFIX Mediator(s) is out of the scope of this specification.

3.4.1.1. Observation Domain Management

In any case, the Observation Domain ID of any IPFIX Message containing Flow Records relevant to no particular Observation Domain, or to multiple Observation Domains, MUST have an Observation Domain ID of 0, as in section 3.1 above, and section 3.1 of [RFC5101].

Internet-Draft <Protocol for IPFIX Mediations> February 2011
IPFIX Mediators that do not change (Options) Template Records
MUST maintain a Template Mapping, as detailed in section 3.2.1,
to ensure that the combination of Observation Domain IDs and
Template IDs do not collide on export.

For IPFIX Mediators that export New (Options) Template Records
unchanged, as in section 3.2.2, there are two options for
Observation Domain ID management. The first and simplest of
these is to completely decouple exported Observation Domain IDs
from received Observation Domain IDs; the IPFIX Mediator, in
this case, comprises its own set of Observation Domain(s)
independent of the Observation Domain(s) of the Original
Exporters.

The second option is to provide or maintain a Template Mapping
for received (Options) Template Records and exported inferred
(Options) Template Records, along with the appropriate
Observation Domain IDs per Transport Session, which ensures that
the combination of Observation Domain IDs and Template IDs do
not collide on export.

In some cases where the IPFIX Message Header can't contain a
consistent Observation Domain for the entire IPFIX Message, but
the Flow Records exported from the IPFIX Mediator should anyway
contain the Observation Domain of the Original Exporter, the
(Options) Template Record must contain the
originalObservationDomainId Information Element. When an IPFIX
Mediator receives Flow Records containing the
originalObservationDomainId Information Element, the IPFIX
Mediator MUST NOT modify its value(s) when composing new Flow
Records with the originalObservationDomainId Information
Element.

3.5. Specific Reporting Requirements

Some specific Options Templates and Options Template Records are
necessary to provide extra information about the Flow Records
and about the Metering Process.

The Options Template Records defined in these subsections, which
impose some constraints on the Metering Process and Exporting
Process implementations in Intermediate Processes, MAY be

Internet-Draft <Protocol for IPFIX Mediations> February 2011
implemented. If implemented, the specific Option Templates
SHOULD be implemented as specified in these subsections.

The minimum set of Information Elements is always specified in
these Specific IPFIX Options Templates. Nevertheless, extra
Information Elements may be used in these specific Options
Templates.

3.5.1. The Flow Keys Options Template

Exactly like the IPFIX protocol [RFC5101], the Flow Keys Option
Template specifies the structure of a Data Record for reporting
the Flow Keys of reported Flows. A Flow Keys Data Record
extends a particular Template Record that is referenced by its
templateId identifier. The Template Record is extended by
specifying which of the Information Elements contained in the
corresponding Data Records describe Flow properties that serve
as Flow Keys of the reported Flow.

The Flow Keys Option Template SHOULD contain the following
Information Elements that are defined in [RFC5102]

templateId	An identifier of a Template. This Information Element MUST be defined as a Scope Field.
------------	---

flowKeyIndicator	Bitmap with the positions of the Flow Keys in the Data Records.
------------------	--

When any Intermediate Process changes the Flow Keys, the Flow
Keys Option Template MUST include the new set of Flow Keys.
Typically, an Intermediate Aggregation Process keeps or reduces
the number of Flow Keys

3.5.2. IPFIX Protocol Options Template

The "Metering Process Statistics Options Template", "The
Metering Process Reliability Statistics Options Template", and
"The Exporting Process Reliability Statistics Options Template",
as specified in [RFC5101], SHOULD be implemented on the IPFIX
Mediator.

Refer to the document specifying a particular Intermediate
Process type for specific values for these Options Template
Records. For example, in case of an Intermediate Aggregation
Process, [IPFIX-MED-AGGR] must specify which values to insert
into the fields of "Metering Process Statistics Options
Template", "The Metering Process Reliability Statistics Options

3.5.3. IPFIX Mediator Options Template

There is no need for a specific Options Template for the IPFIX Mediator; instead, each Intermediate Process type requires some particular metadata. For example, a specification of IPFIX flow anonymisation including an Options Template for the export of metadata about anonymised flows is described in [IPFIX-MED-ANON]; when anonymising Flows Records, IPFIX Mediators SHOULD add the Options Template specified therein to annotate the exported dataTransport Session Management Sctp [RFC4960] using the PR-SCTP extension specified in [RFC3758] MUST be implemented by all compliant IPFIX Mediator implementations. UDP [UDP] MAY also be implemented by compliant IPFIX Mediator implementations. TCP [TCP] MAY also be implemented by IPFIX Mediator compliant implementations.

PR-SCTP SHOULD be used in deployments where IPFIX Mediators and Collectors are communicating over links that are susceptible to congestion. PR-SCTP is capable of providing any required degree of reliability.

TCP MAY be used in deployments where IPFIX Mediators and Collectors communicate over links that are susceptible to congestion, but PR-SCTP is preferred due to its ability to limit back pressure on Exporters and its message versus stream orientation.

UDP MAY be used, although it is not a congestion-aware protocol. However, the IPFIX traffic between IPFIX Mediator and Collector MUST run in an environment where IPFIX traffic has been provisioned for, or is contained through some other means.

3.6. The Collecting Process's Side

An IPFIX Mediator MUST produce IPFIX Messages understandable by a RFC5101-compliant IPFIX Collector, with the additional specification in the IPFIX Structured Data [IPFIX-STRUCT].

Therefore the Collecting Process on the top Collector MUST support the IPFIX protocol [RFC5101] and the IPFIX Structured Data [IPFIX-STRUCT].

In some cases such as an Intermediate Aggregation Process aggregating Flow Records from multiple Original Exporters, a consistent configuration of the Metering Processes and Exporting Processes on these Original offers some advantages. For example, consistent active timeout, inactive timeout, and/or consistent export time allows to compare the number of the Flow Records per period of time. For example, consistent Sampling algorithm and parameters might allow to compare Flow Records accuracy.

While this is tempting to include all configuration parameters in Flow Records for the IPFIX Mediator to draw its own conclusion, the consistency of the configuration should be verified out of band, with the MIB modules ([RFC5815] and [PSAMP-MIB] or with the Configuration Data Model for IPFIX and PSAMP [IPFIX-CONF]

4. New Information Elements

EDITOR NOTE: please change the TBD1, TBD2, and TBD3, with the IANA newly assigned numbers.

4.1. - originalExporterIPv4Address

Description: The IPv4 address used by the Exporting Process on the Original Exporter. This is used by an IPFIX Mediator Exporting Process to identify the Original Exporter.

Abstract Data Type: ipv4Address

ElementId: TBD3

Status: Proposed

4.2. originalExporterIPv6Address

Description: The IPv6 address used by the Exporting Process on the Original Exporter. This is used by the IPFIX Mediator Exporting Process to identify the Original Exporter.

Abstract Data Type: ipv6Address

Internet-Draft <Protocol for IPFIX Mediations> February 2011
 ElementId: TBD2

 Status: Proposed

4.3. originalObservationDomainId

Description: An identifier of the Observation Domain on the Original Exporter, where the metered IP packets are observed. This is used by the IPFIX Mediator Exporting Process to identify an Observation Domain as received from the Original Exporter.

Abstract Data Type: unsigned32

ElementId: TBD3

Status: Proposed

5. Security Considerations

The same security considerations as for the IPFIX Protocol [RFC5101] apply.

As they act as both IPFIX Collecting Processes and Exporting Processes, the Security Considerations for IPFIX [RFC5101] apply as well to Mediators. The Security Considerations for IPFIX Files [RFC5655] apply as well to IPFIX Mediators that write IPFIX Files or use them for internal storage. However, there are a few specific considerations that IPFIX Mediator implementations must take into account in addition.

By design, IPFIX Mediators are "men-in-the-middle": they intercede in the communication between an Original Exporter (or another upstream Mediator) and a downstream Collecting Process. This has two important implications for the level of confidentiality provided across an IPFIX Mediator, and the ability to protect data integrity and Original Exporter authenticity across a Mediator. These are addressed in more detail in the Security Considerations for Mediators in [IPFIX-MED-FMWK].

Note that, while Mediators can use the exporterCertificate and collectorCertificate Information Elements defined in [RFC5655] as described in section 9.3 of [IPFIX-MED-FMWK] to export

Internet-Draft <Protocol for IPFIX Mediations> February 2011
information about X.509 identities in upstream TLS-protected
Transport Sessions, this mechanism cannot be used to provide
true end-to-end assertions about a chain of IPFIX Mediators: any
Mediator in the chain can simply falsify the information about
upstream Transport Sessions. In situations where information
about the chain of mediation is important, it must be determined
out of band.

6. IANA Considerations

This document specifies three new IPFIX Information Elements: the
applicationDescription, applicationTag and the applicationName.

New Information Elements to be added to the IPFIX Information
Element registry at [IANA-IPFIX] are listed below.

EDITOR'S NOTE: the XML specification in Appendix A must be updated
with the elementID values allocated, i.e. TBD1, TBD2, and TBD3,
must be replaced.

6.1. originalExporterIPv4Address

Name: originalExporterIPv4Address

Description:

The IPv4 address used by the Exporting Process on the Original
Exporter. This is used by an IPFIX Mediator Exporting Process
to identify the Original Exporter.

Abstract Data Type: ipv4Address

Data Type Semantics: identifier

ElementId: TBD1

Status: current

6.2. originalExporterIPv6Address

Name: originalExporterIPv6Address

Description:

The IPv6 address used by the Exporting Process on the Original
Exporter. This is used by the IPFIX Mediator Exporting Process
to identify the Original Exporter.

Abstract Data Type: ipv6Address

Data Type Semantics: identifier

ElementId: TBD2

Status: current

Name: originalObservationDomainId

Description:

 An identifier of the Observation Domain on the Original
 Exporter, where the metered IP packets are observed. This is
 used by the IPFIX Mediator Exporting Process to identify an
 Observation Domain as received from the Original Exporter.

Abstract Data Type: unsigned32

Data Type Semantics: identifier

ElementId: TBD3

Status: current

7. References

7.1. Normative References

- [RFC2119] S. Bradner, Key words for use in RFCs to Indicate Requirement Levels, BCP 14, RFC 2119, March 1997
- [RFC3758] Stewart, R., Ramalho, M, Xie, Q., Tuexen, M., and P. Conrad, "Stream Control Transmission Protocol (SCTP), Partial Reliability Extension", May 2004
- [RFC4960] Stewart, R., Ed., "Stream Control Transmission Protocol", RFC 4960, September 2007.
- [RFC5101] Claise, B., Ed., "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information", RFC 5101, January 2008.
- [RFC5102] Quittek, J., Bryant, S., Claise, B., Aitken, P., and J. Meyer, "Information Model for IP Flow Information Export", RFC 5102, January 2008.
- [RFC5655] Trammell, B., Boschi, E., Mark, L., Zseby, T., and A. Wagner, "Specification of the IP Flow Information Export (IPFIX) File Format", RFC 5655, October 2009.
- [RFC5815] Dietz, T., Kobayashi, A., Claise, B., and G. Muenz, "Definitions of Managed Objects for IP Flow Information Export", RFC 5815, April 2010.
- [IPFIX-MED-FLOWSEL] Peluso, L., Zseby, T., D'antonio, S., and M. Molina, "Flow Selection Techniques", draft-ietf-ipfix-

Internet-Draft <Protocol for IPFIX Mediations> February 2011
flow-selection-tech-04.txt, Internet-Draft work in
progress, January 2011.

[IPFIX-MED-AGGR] Trammell, B., Boschi, E., and A. Wagner,
"Exporting Aggregated Flow Data using the IP Flow
Information Export (IPFIX) Protocol", draft-trammell-
ipfix-a9n-01.txt, Internet-Draft work in progress,
October 2010.

[IPFIX-STRUCT] Claise, B., Dhandapani, G., Aitken, P., and S.
Yates, "Export of Structured Data in IPFIX", draft-
ietf-ipfix-structured-data-04.txt, Internet-Draft work
in progress, December 2010.

[PSAMP-MIB] Dietz, T., Claise, B., and J. Quittek "Definitions
of Managed Objects for Packet Sampling", draft-ietf-
ipfix-psamp-mib-02.txt, Internet-Draft work in
progress, November 2010.

[IPFIX-CONF] Muenz, G., Claise, B., and P. Aitken "Configuration
Data Model for IPFIX and PSAMP", draft-ietf-ipfix-
configuration-model-08, Internet-Draft work in
progress, October 2010.

7.2. Informative References

[TCP] Postel, J., "Transmission Control Protocol", STD 7, RFC
793, September 1981.

[UDP] Postel, J., "User Datagram Protocol", STD 6, RFC 768,
August 1980.

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

[RFC3954] Claise, B. (Ed), "Cisco Systems NetFlow Services
Export Version 9", RFC 3954, October 2004

[RFC5470] Sadasivan, G., Brownlee, N., Claise, B., and J.
Quittek, "Architecture Model for IP Flow Information
Export", RFC5470, March 2009

Internet-Draft <Protocol for IPFIX Mediations> February 2011
[RFC5472] Zseby, T., Boschi, E., Brownlee, N., and B. Claise,
 "IP Flow Information Export (IPFIX) Applicability", RFC
 5472, March 2009

[RFC5476] Claise, B., Quittek, J., and A. Johnson, "Packet
 Sampling (PSAMP) Protocol Specifications", RFC 5476,
 March 2009.

[RFC5982] Kobayashi, A. (Ed), Claise, B. (Ed), "P Flow
 Information Export (IPFIX) Mediation: Problem
 Statement", RFC 5982, August 2010.

[IPFIX-MED-FMWK] Kobayashi, A., Claise, B., and K. Ishibashi,
 "IPFIX Mediation: Framework", draft-ietf-ipfix-
 mediators-framework-09, Internet-Draft work in
 progress, October 2010.

[IPFIX-MED-ANON] Boschi, E., Trammell, B. "IP Flow Anonymisation
 Support", draft-ietf-ipfix-anon-06.txt, Internet-Draft
 work in progress, January 2011.

[IANA-IPFIX] <http://www.iana.org/assignments/ipfix/ipfix.xhtml>

8. Author's Addresses

Benoit Claise
Cisco Systems, Inc.
De Kleetlaan 6a b1
Diegem 1813
Belgium

Phone: +32 2 704 5622
Email: bclaise@cisco.com

Atsushi Kobayashi
NTT Information Sharing Platform Laboratories
3-9-11 Midori-cho
Musashino-shi, Tokyo 180-8585
Japan

Phone: +81-422-59-3978
Email: akoba@nttv6.net
URI: <http://www3.plala.or.jp/akoba/>

Internet-Draft <Protocol for IPFIX Mediations> February 2011
Brian Trammell
ETH Zurich
Gloriastrasse 35
8092 Zurich
Switzerland

Phone: +41 44 632 70 13
EMail: trammell@tik.ee.ethz.ch

9. Appendix A. Additions to XML Specification of IPFIX Information Elements

This appendix contains additions to the machine-readable description of the IPFIX information model coded in XML in Appendix A and Appendix B in [RFC5102]. Note that this appendix is of informational nature, while the text in Section 6. (generated from this appendix) is normative.

The following field definitions are appended to the IPFIX information model in Appendix A of [RFC5102].

```
<field name="originalExporterIPv4Address"
      dataType="ipv4Address"
      group="config"
      elementId="TBD1" applicability="all" status="current">
  <description>
    <paragraph>
      The IPv4 address used by the Exporting Process on the
      Original Exporter. This is used by an IPFIX Mediator
      Exporting Process to identify the Original Exporter.
    </paragraph>
  </description>
</field>
```

```
<field name="originalExporterIPv6Address"
      dataType="ipv6Address"
      group="config"
      elementId="TBD2" applicability="all" status="current">
  <description>
    <paragraph>
      The IPv6 address used by the Exporting Process on the
      Original Exporter. This is used by the IPFIX Mediator
      Exporting Process to identify the Original Exporter.
    </paragraph>
  </description>
</field>
```

```
Internet-Draft      <Protocol for IPFIX Mediations>  February 2011
  <field name="originalObservationDomainId"
    dataType="unsigned32"
    group="config"
    elementId="TBD3" applicability="all" status="current">
    <description>
      <paragraph>
        An identifier of the Observation Domain on the Original
        Exporter, where the metered IP packets are observed.
        This is used by the IPFIX Mediator Exporting Process to
        identify an Observation Domain as received from the
        Original Exporter.
      </paragraph>
    </description>
  </field>
```


IP Flow Information Export
Internet-Draft
Intended status: Standards Track
Expires: September 16, 2011

S. Kashima
K. Nakata
NTT
A. Kobayashi
NTT East
March 15, 2011

Information Elements for Data Link Layer Traffic Measurement
draft-kashima-ipfix-data-link-layer-monitoring-05

Abstract

This document describes Information Elements related to data link layer. They are used by the IP Flow Information Export (IPFIX) protocol for encoding measured data link layer traffic information.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on September 16, 2011.

Copyright Notice

Copyright (c) 2011 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

This document may contain material from IETF Documents or IETF Contributions published or made publicly available before November 10, 2008. The person(s) controlling the copyright in some of this material may not have granted the IETF Trust the right to allow modifications of such material outside the IETF Standards Process. Without obtaining an adequate license from the person(s) controlling the copyright in such materials, this document may not be modified outside the IETF Standards Process, and derivative works of it may not be created outside the IETF Standards Process, except to format it for publication as an RFC or to translate it into languages other than English.

Table of Contents

1. Introduction	4
1.1. Conventions Used in This Document	4
2. Extended Ethernet Technology	4
2.1. Wide-Area Ethernet Network Summary	4
2.2. Data Center Network Summary	5
2.3. Multiple Path Ethernet Summary	5
3. New Information Elements	6
3.1. dataLinkFrameSize	6
3.2. dataLinkFrameSection	7
3.3. dataLinkFrameType	7
3.4. sectionOffset	8
3.5. sectionObservedOctets	8
3.6. dot1qServiceInstanceTag	9
3.7. dot1qServiceInstanceId	9
3.8. dot1qServiceInstancePriority	9
3.9. dot1qCustomerDestinationMacAddress	10
3.10. dot1qCustomerSourceMacAddress	10
4. Modification of Existing Information Elements Related to Packet Section	11
4.1. ipHeaderPacketSection	11
4.2. ipPayloadPacketSection	11
4.3. mplsLabelStackSection	11
4.4. mplsPayloadPacketSection	12
5. Modification of Existing Information Elements Related to VLAN Tag	12
5.1. dot1qVlanId	12
5.2. dot1qPriority	13
5.3. dot1qCustomerVlanId	13
5.4. dot1qCustomerPriority	14
6. Security Considerations	15
7. IANA Considerations	15
8. References	15
8.1. Normative References	15
8.2. Informative References	15
Appendix A. Frame Formats in Wide-Area Ethernet Network	17
Appendix B. Frame Formats in Data Center Network	22
Appendix C. Frame Format in TRILL	24
Appendix D. Template Formats Example	25
Authors' Addresses	25

1. Introduction

Ethernet [IEEE802.1D] and VLAN (Virtual LAN) [IEEE802.1Q-2005] technologies used to be used only in Local Area Networks. Recently, they have been used in Wide Area Networks, e.g., L2-VPN services. Accordingly, the IEEE802.1Q standard has been enhanced to IEEE802.1ad [IEEE802.1ad-2005] and IEEE802.1ah [IEEE802.1ah-2008]. And, Ethernet in data center also has been enhanced for server virtualization and I/O consolidation.

While these innovations provide flexibility, scalability, and mobility to an existing network architecture, it increases the complexity of traffic measurement due to the existence of various Ethernet header formats. To cope with this, a more sophisticated method is required.

IPFIX/PSAMP helps to resolve these problems. However, the PSAMP Information Model [RFC5477] and the IPFIX Information Model [RFC5101] are not yet enough for Information Elements related to data link layer, e.g., Ethernet header forms. This document describes the Information Elements related to data link layers that enable a more sophisticated traffic measurement method.

1.1. Conventions Used in This Document

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 RFC 2119 [RFC2119].

2. Extended Ethernet Technology

2.1. Wide-Area Ethernet Network Summary

Provider Bridge [IEEE802.1ad-2005] and Provider Backbone Bridge [IEEE802.1ah-2008], which are standards for the Wide-Area Ethernet, are described below.

- o In Provider Bridge [IEEE802.1ad-2005], there are two VLAN IDs: Service VLAN Identifier (S-VID) and Customer VLAN Identifier (C-VID). S-VID is assigned to an Ethernet frame by a service provider, while C-VID is independently assigned to an Ethernet frame by a customer. Frame switching in a service provider network is based on only S-VID.
- o In Provider Backbone Bridge [IEEE802.1ah-2008], new Ethernet fields, such as Backbone VLAN Identifier (B-VID) and Backbone Service Instance Identifier (I-SID), are introduced to overcome the limitations on the VLAN identifier space on IEEE802.1ad

[IEEE802.1ad-2005] and to isolate the service provider and customer identifier spaces. Frame switching is based on a 12-bit B-VID, and customer identification is based on a 24-bit I-SID. A flexible network design has become possible because network management is separated from customer management. Other Ethernet fields that indicate quality of service (QoS) class are B-PCP, B-DEI, I-PCP, and I-DEI.

Provider Backbone Bridge enables a wide-area Ethernet service to be improved from a flat network to a hierarchical network co-existing Provider Bridge and Provider Backbone Bridge.

Frame formats used in Wide-Area Ethernet are shown in Appendix A.

2.2. Data Center Network Summary

In data center networks, Ethernet needs to be enhanced to provide the flexibility, mobility for server virtualization, and I/O consolidation. In IEEE802.1 Data Center Bridging Task Group, several Ethernet header formats are proposed to enable a simplifying networks and server managements.

The one of the enhanced methods is Bridge Port Extension [IEEE802.1Qbh], which brings a traffic exchange point to upper bridges. Bridge Port Extension introduces a Ethernet format named Extension Tag (E-TAG) in addition to existing Service VLAN Tag (S-TAG) and Customer VLAN Tag (C-TAG) to move the policy enhancement to upper bridges in data center network. On the other hand, the complexity for traffic measurement would be increased, because multiple Ethernet header formats as shown in Appendix B co-exist in the same network.

2.3. Multiple Path Ethernet Summary

Inside and between data center networks, Ethernet need to be enhanced to support multiple path for effective utilization of bandwidth. Existing Ethernet does not support multiple path because it is based on STP (Spanning Tree Protocol) [IEEE802.1D].

Two solutions for multiple path Ethernet are discussed for standardization. One is Shortest Path Bridging [IEEE802.1aq] in IEEE 802, the other is TRILL (Transparent Interconnection of Lots of Links) [I-D.ietf-trill-rbridge-protocol] in IETF. Both solutions realize multiple path with IS-IS [ISO_IEC.10589_2002] and Layer 2 over Layer 2 encapsulation. Shortest Path Bridging uses IEEE802.1ah [IEEE802.1ah-2008] header as an encapsulation technology and TRILL uses an unique header as shown in Appendix C.

In multiple path Ethernet network, a traffic measurement based on MAC address and VLAN Tag is more important than single path Ethernet network in which traffic measurement is based on only VLAN Tag.

3. New Information Elements

The following Information Elements are necessary for enabling the IPFIX/PSAMP traffic measurement for data link layer, which is not limited to Ethernet because the method can be applied to other data link protocols as well. Note that these are proposed IDs, subject to approval by IANA.

ID	Name
312	dataLinkFrameSize
315	dataLinkFrameSection
347	dataLinkFrameType
348	sectionOffset
349	sectionObservedOctets
350	dot1qServiceInstanceTag
351	dot1qServiceInstanceId
352	dot1qServiceInstancePriority
353	dot1qCustomerDestinationMacAddress
354	dot1qCustomerSourceMacAddress

3.1. dataLinkFrameSize

Description:

This Information Element specifies the length of the selected data link frame.

The data link layer is defined in [ISO_IEC.7498-1_1994].

Abstract Data Type: unsigned16

Data Type Semantics: quantity

ElementId: 312

Status: current

3.2. dataLinkFrameSection

Description:

This Information Element carries `n` octets from the data link frame of a selected frame, starting `sectionOffset` octets into the frame.

The `sectionObservedOctets` expresses how much data was observed, while the remainder is padding.

When the `sectionObservedOctets` field corresponding to this Information Element exists, this Information Element MAY have a fixed length and MAY be padded, or MAY have a variable length.

When the `sectionObservedOctets` field corresponding to this Information Element does not exist, this Information Element SHOULD have a variable length and MUST NOT be padded. In this case, the size of the exported section may be constrained due to limitations in the IPFIX protocol.

Further Information Elements, i.e., `dataLinkFrameType` and `dataLinkFrameSize` are needed to specify the data link type and the size of the data link frame of this Information Element. A set of these Information Elements MAY be contained in a structured data type, as expressed in [I-D.ietf-ipfix-structured-data]. Or a set of these Information Elements MAY be contained in one Flow Record as shown in Appendix D.

The data link layer is defined in [ISO_IEC.7498-1_1994].

Abstract Data Type: `octetArray`

ElementId: 315

Status: current

3.3. dataLinkFrameType

Description:

This Information Element specifies the type of the selected data link frame.

The following data link types are defined here.

- 0x01 ETHERNET

Further values may be assigned by IANA.

The data link layer is defined in [ISO_IEC.7498-1_1994].

Abstract Data Type: unsigned16

Data Type Semantics: identifier

ElementId: 347

Status: current

3.4. sectionOffset

Description:

This Information Element specifies the offset of the packet section (e.g., dataLinkFrameSection, ipHeaderPacketSection, ipPayloadPacketSection, mplsLabelStackSection and mplsPayloadPacketSection). If this Information Element is omitted, it defaults to zero.

Abstract Data Type: unsigned16

Data Type Semantics: quantity

ElementId: 348

Status: current

3.5. sectionObservedOctets

Description:

This Information Element specifies the observed length of the packet section (e.g., dataLinkFrameSection, ipHeaderPacketSection, ipPayloadPacketSection, mplsLabelStackSection and mplsPayloadPacketSection) when padding is used.

The packet section may be of a fixed size larger than the sectionObservedOctets. In this case, octets in the packet section beyond the sectionObservedOctets MUST follow the [RFC5101] rules for padding (ie, be composed of zero (0) valued octets).

Abstract Data Type: unsigned16

Data Type Semantics: quantity

ElementId: 349

Status: current

3.6. dot1qServiceInstanceTag

Description:

This Information Element, which may have 16 octets length, carries the Backbone Service Instance Tag (I-TAG) Tag Control Information (TCI) field of an Ethernet frame as described in [IEEE802.1ah-2008].

Abstract Data Type: octetArray

Data Type Semantics: identifier

ElementId: 350

Status: current

3.7. dot1qServiceInstanceId

Description:

The value of the 3-bit Backbone Service Instance Priority Code Point (I-PCP) portion of the Backbone Service Instance Tag (I-TAG) Tag Control Information (TCI) field of an Ethernet frame as described in section 9.8 of [IEEE802.1ah-2008]. The structure and semantics within the Tag Control Information field are defined in IEEE802.1ah.

Abstract Data Type: unsigned32

Data Type Semantics: identifier

ElementId: 351

Status: current

3.8. dot1qServiceInstancePriority

Description:

The value of the 24-bit Backbone Service Instance Identifier (I-SID) portion of the Backbone Service Instance Tag (I-TAG) Tag Control Information (TCI) field of an Ethernet frame as described in section 9.8 of [IEEE802.1ah-2008]. The structure and semantics

within the Tag Control Information field are defined in IEEE802.1ah.

Abstract Data Type: unsigned8

Data Type Semantics: identifier

ElementId: 352

Status: current

3.9. dot1qCustomerDestinationMacAddress

Description:

The value of the Customer Destination Address (C-DA) portion of the Backbone Service Instance Tag (I-TAG) Tag Control Information (TCI) field of an Ethernet frame as described in section 9.8 of [IEEE802.1ah-2008]. The structure and semantics within the Tag Control Information field are defined in IEEE802.1ah.

Abstract Data Type: macAddress

Data Type Semantics: identifier

ElementId: 353

Status: current

3.10. dot1qCustomerSourceMacAddress

Description:

The value of the Customer Source Address (C-SA) portion of the Backbone Service Instance Tag (I-TAG) Tag Control Information (TCI) field of an Ethernet frame as described in section 9.8 of [IEEE802.1ah-2008]. The structure and semantics within the Tag Control Information field are defined in IEEE802.1ah.

Abstract Data Type: macAddress

Data Type Semantics: identifier

ElementId: 354

Status: current

4. Modification of Existing Information Elements Related to Packet Section

--- This is open issue. ---

New Information Elements related to packet section (ie, sectionOffset and sectionObservedOctets) can be applied to not only dataLinkFrameSection but also all kinds of packet section. In this case, existing Information Elements Description should be modified as follows:

4.1. ipHeaderPacketSection

Description:

TBD

Abstract Data Type: octetArray

ElementId: 313

Status: current

4.2. ipPayloadPacketSection

Description:

TBD

Abstract Data Type: octetArray

ElementId: 314

Status: current

4.3. mplsLabelStackSection

Description:

TBD

Abstract Data Type: octetArray

ElementId: 316

Status: current

4.4. mplsPayloadPacketSection

Description:

TBD

Abstract Data Type: octetArray

ElementId: 317

Status: current

5. Modification of Existing Information Elements Related to VLAN Tag

Information Elements related to Backbone Service Instance Tag (I-TAG) and Backbone VLAN Tag (B-TAG) are required in order to monitor IEEE802.1ah traffic with IPFIX/PSAMP. Because I-TAG has different structure and semantics from Service VLAN Tag (S-TAG) and Customer VLAN Tag (C-TAG), new Information Elements related to I-TAG are added in section 3. But because B-TAG has same different structure and semantics with C-TAG and S-TAG, Information Elements related to B-TAG reuse existing Information Elements related to C-TAG and S-TAG. Though they reuse existing Information Elements, it required to modify existing Descriptions and Reference as follows:

5.1. dot1qVlanId

Description:

The value of the 12-bit VLAN Identifier portion of the Tag Control Information field of an Ethernet frame as described in section 3.5.5 of [IEEE802.3-2005]. The structure and semantics within the Tag Control Information field are defined in IEEE P802.1Q. In case of a QinQ frame it represents the outer tag's VLAN identifier, in case of an IEEE 802.1ad frame it represents the Service VLAN identifier in the S-TAG Tag Control Information (TCI) field as described in [IEEE802.1ad-2005] and in case of an IEEE 802.1ah frame it represents the Backbone VLAN identifier in the B-TAG Tag Control Information (TCI) field as described in [IEEE802.1ah-2008].

Abstract Data Type: unsigned16

Data Type Semantics: identifier

ElementId: 243

Status: current

Reference:

- (1) [IEEE802.3-2005]
- (2) [IEEE802.1ad-2005]
- (3) [IEEE802.1ah-2008]

5.2. dot1qPriority

Description:

The value of the 3-bit User Priority portion of the Tag Control Information field of an Ethernet frame as described in section 3.5.5 of [IEEE802.3-2005]. The structure and semantics within the Tag Control Information field are defined in IEEE P802.1Q. In case of a QinQ frame it represents the outer tag's 3-bit Class of Service (CoS) identifier, in case of an IEEE 802.1ad frame it represents the 3-bit Priority Code Point (PCP) portion of the S-TAG Tag Control Information (TCI) field as described in [IEEE802.1ad-2005] and in case of an IEEE 802.1ah frame it represents the 3-bit Priority Code Point (PCP) portion of the B-TAG Tag Control Information (TCI) field as described in [IEEE802.1ah-2008].

Abstract Data Type: unsigned8

Data Type Semantics: identifier

ElementId: 244

Status: current

Reference:

- (1) [IEEE802.3-2005]
- (2) [IEEE802.1ad-2005]
- (3) [IEEE802.1ah-2008]

5.3. dot1qCustomerVlanId

Description:

In case of a QinQ frame it represents the inner tag's (*) VLAN identifier, in case of an IEEE 802.1ad frame it represents the Customer VLAN identifier in the C-TAG Tag Control Information (TCI) field as described in [IEEE802.1ad-2005] and in case of an IEEE 802.1ah frame it represents the Customer VLAN identifier in

the C-TAG Tag Control Information (TCI) field in encapsulated IEEE 802.1ad or IEEE 802.1Q frame as described in [IEEE802.1ah-2008].

(*) Note: the 802.1Q tag directly following the outer one.

Abstract Data Type: unsigned16

Data Type Semantics: identifier

ElementId: 245

Status: current

Reference:

- (1) [IEEE802.1ad-2005]
- (2) [IEEE802.1Q-2005]
- (3) [IEEE802.1ah-2008]

5.4. dot1qCustomerPriority

Description:

In case of a QinQ frame it represents the inner tag's (*) Class of Service (CoS) identifier, in case of an IEEE 802.1ad frame it represents the 3-bit Priority Code Point (PCP) portion of the C-TAG Tag Control Information (TCI) field as described in [IEEE802.1ad-2005] and in case of an IEEE 802.1ah frame it represents the 3-bit Priority Code Point (PCP) portion of the C-TAG Tag Control Information (TCI) field in encapsulated IEEE 802.1ad or IEEE 802.1Q frame as described in [IEEE802.1ad-2005].

(*) Note: the 802.1Q tag directly following the outer one.

Abstract Data Type: unsigned8

Data Type Semantics: identifier

ElementId: 246

Status: current

Reference:

- (1) [IEEE802.1ad-2005]
- (2) [IEEE802.1Q-2005]

(3) [IEEE802.1ah-2008]

6. Security Considerations

The recommendations in this document do not introduce any additional security issues to those already mentioned in [RFC5101] and [RFC5477].

7. IANA Considerations

This document requests that the Information Element IDs are allocated as shown in section 3

In addition, the dataLinkFrameType Information Element requires the creation of new IANA registries.

And this document requests that the existing Information Element Description are modified as shown in section 4 and 5

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

8.2. Informative References

[I-D.ietf-ipfix-structured-data]
Claise, B., Dhandapani, G., Yates, S., and P. Aitken,
"Export of Structured Data in IPFIX",
draft-ietf-ipfix-structured-data-05 (work in progress),
March 2011.

[I-D.ietf-trill-rbridge-protocol]
3rd, D., Dutt, D., Gai, S., Ghanwani, A., and R. Perlman,
"Rbridges: Base Protocol Specification",
draft-ietf-trill-rbridge-protocol-16 (work in progress),
March 2010.

[IEEE802.1D]
IEEE Computer Society, "IEEE Standards for Local and
Metropolitan Area Networks: Media Access Control (MAC)
Bridges", IEEE Std 802.1D-2004, June 2004.

[IEEE802.1Q-2005]

IEEE Computer Society, "IEEE Standards for Local and Metropolitan Area Networks: Virtual Bridged Local Area Networks", IEEE Std 802.1Q-2005, May 2006.

[IEEE802.1Qbh]

IEEE Computer Society, "Draft Standards for Local and Metropolitan Area Networks: Virtual Bridged Local Area Networks Amendment: Bridge Port Extension", IEEE Std P802.1Qbh/D1.1, February 2011.

[IEEE802.1ad-2005]

IEEE Computer Society, "IEEE Standards for Local and Metropolitan Area Networks: Virtual Bridged Local Area Networks Amendment 4: Provider Bridges", IEEE Std 802.1ad-2005, May 2006.

[IEEE802.1ah-2008]

IEEE Computer Society, "IEEE Standards for Local and Metropolitan Area Networks: Virtual Bridged Local Area Networks Amendment 7: Provider Backbone Bridges", IEEE Std 802.1ah-2008, August 2008.

[IEEE802.1aq]

IEEE Computer Society, "Draft Standards for Local and Metropolitan Area Networks: Virtual Bridged Local Area Networks Amendment: Shortest Path Bridging", IEEE Std P802.1aq/D3.6, February 2011.

[IEEE802.3-2005]

IEEE Computer Society, "IEEE Standard for Information Technology - Telecommunications and Information Exchange Between Systems - Local and Metropolitan Area Networks - Specific Requirements Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications", IEEE Std 802.3-2005, December 2005.

[ISO_IEC.10589_2002]

International Organization for Standardization, "Information technology -- Telecommunications and information exchange between systems -- Intermediate System to Intermediate System intra-domain routing information exchange protocol for use in conjunction with the protocol for providing the connectionless-mode network service (ISO 8473)", ISO Standard 10589:2002, November 2002.

- [ISO_IEC.7498-1_1994] International Organization for Standardization, "Information technology -- Open Systems Interconnection -- Basic Reference Model: The Basic Mode", ISO Standard 7498-1:1994, June 1996.
- [RFC2863] McCloghrie, K. and F. Kastenholz, "The Interfaces Group MIB", RFC 2863, June 2000.
- [RFC3954] Claise, B., "Cisco Systems NetFlow Services Export Version 9", RFC 3954, October 2004.
- [RFC5101] Claise, B., "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information", RFC 5101, January 2008.
- [RFC5475] Zseby, T., Molina, M., Duffield, N., Niccolini, S., and F. Raspall, "Sampling and Filtering Techniques for IP Packet Selection", RFC 5475, March 2009.
- [RFC5477] Dietz, T., Claise, B., Aitken, P., Dressler, F., and G. Carle, "Information Model for Packet Sampling Exports", RFC 5477, March 2009.

Appendix A. Frame Formats in Wide-Area Ethernet Network

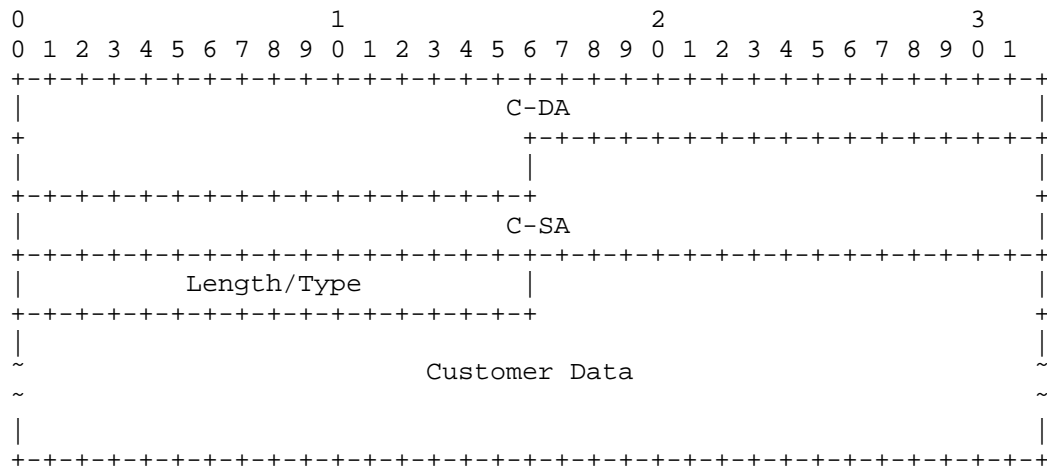


Figure A-1: IEEE802.1D Frame Format in Customer Bridged Network

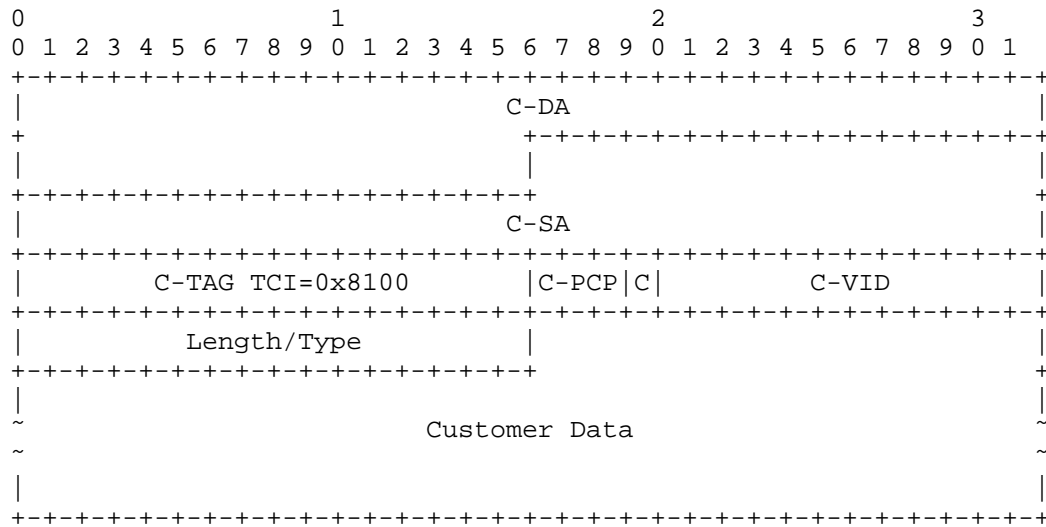


Figure A-2: IEEE802.1Q Frame Format in Customer Bridged Network

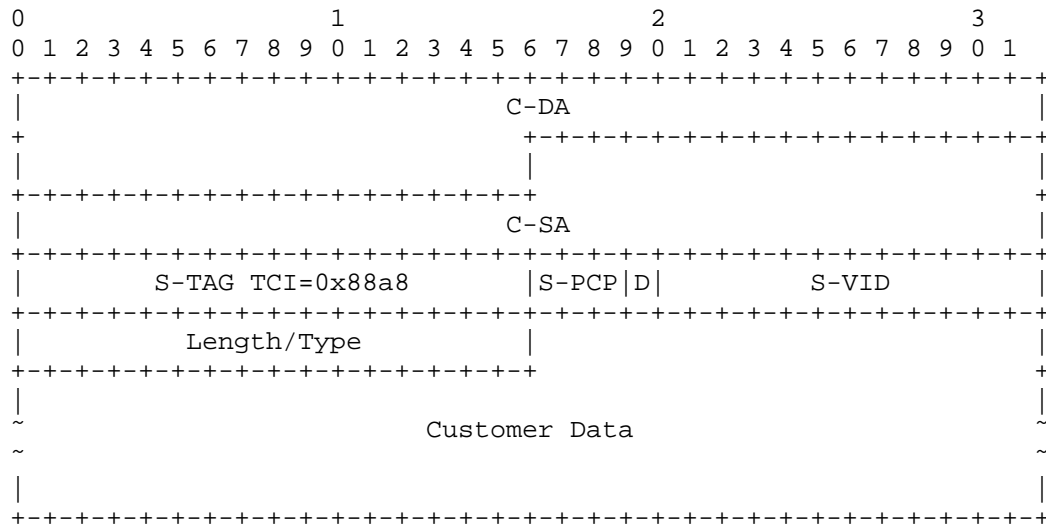


Figure A-3: IEEE802.1ad (no C-Tag) Frame Format in Provider Bridged Network

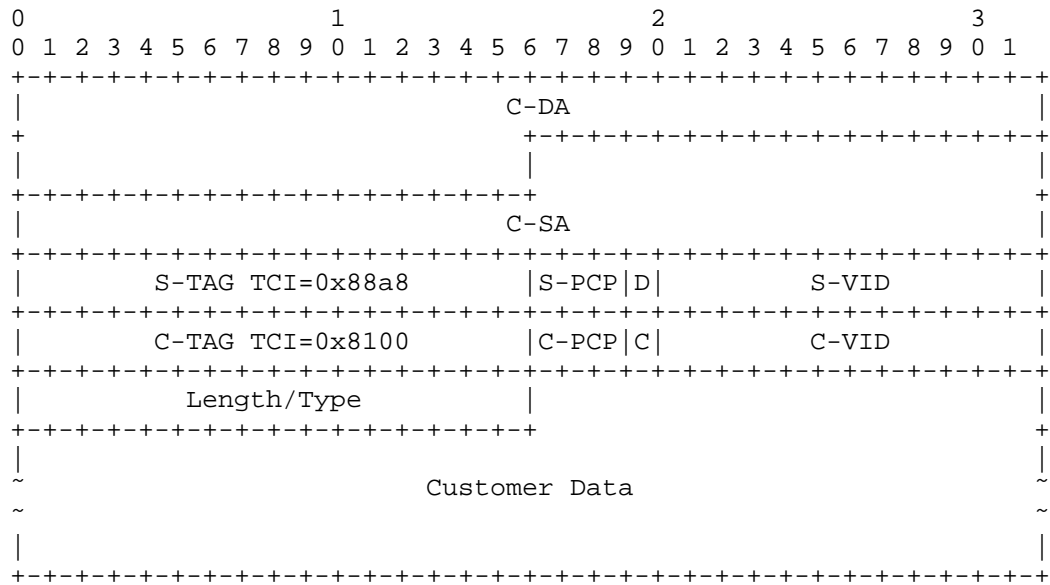


Figure A-4: IEEE802.1ad (C-Tagged) Frame Format in Provider Bridged Network

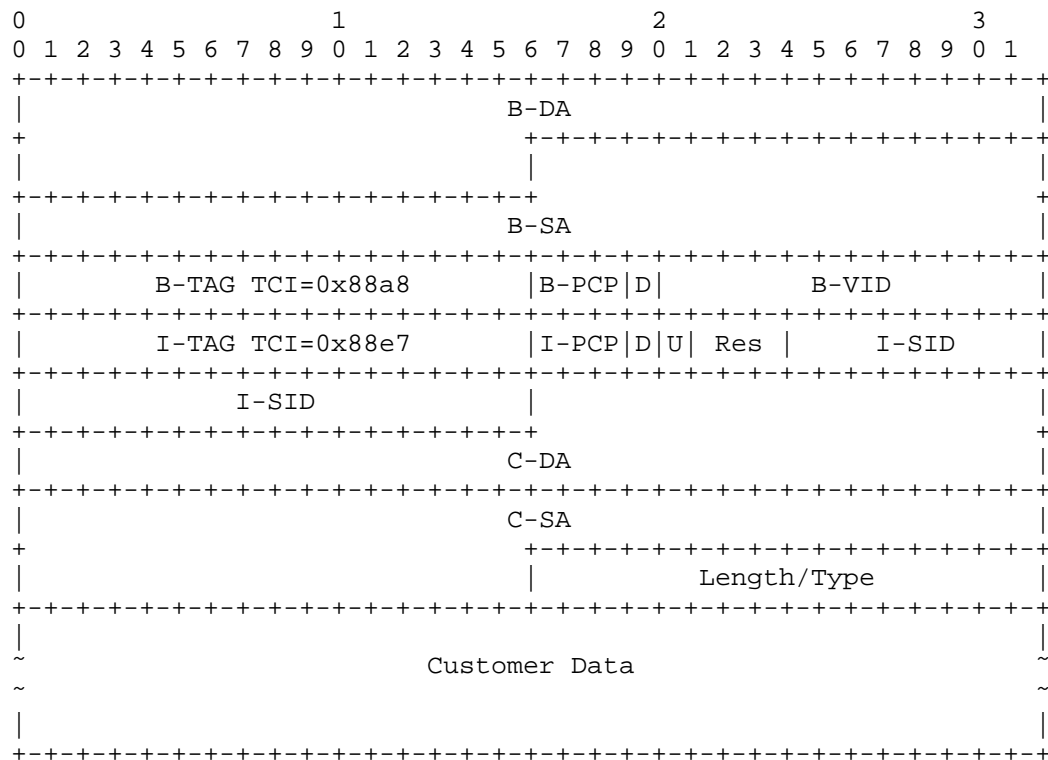


Figure A-5: IEEE802.1ah (no C-Tag) Frame Format in Provider Backbone Bridged Network

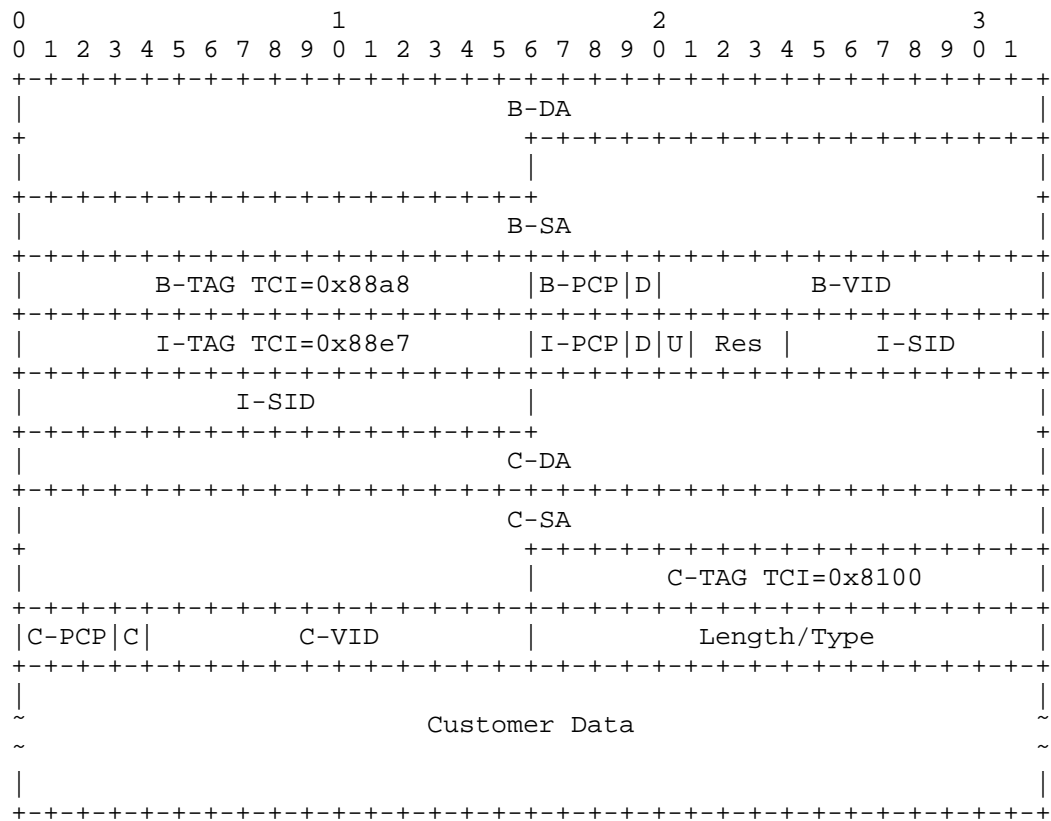


Figure A-6: IEEE802.1ah (C-Tagged) Frame Format in Provider Backbone Bridged Network

Appendix B. Frame Formats in Data Center Network

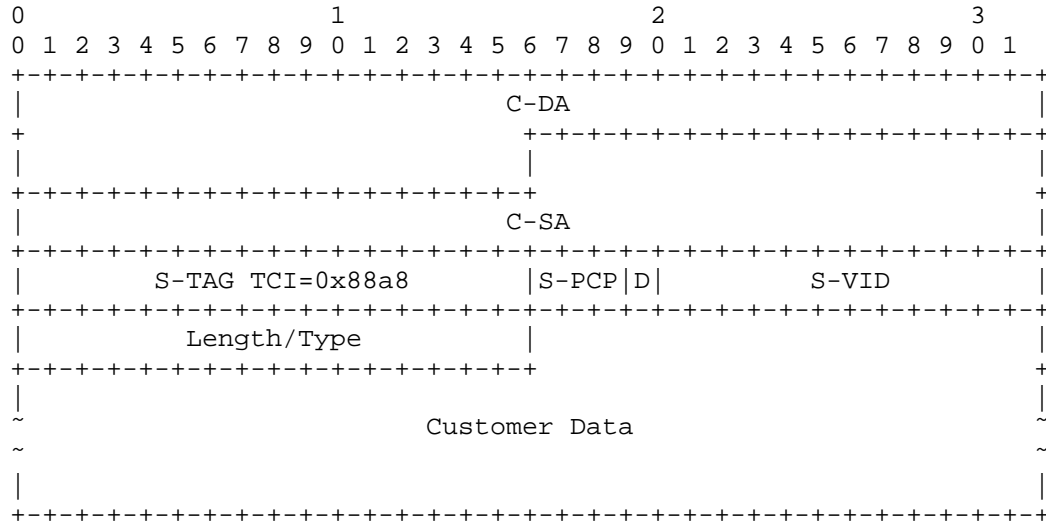


Figure B-1: IEEE802.1Qbh (S-TAG) Frame Format in Data Center Network

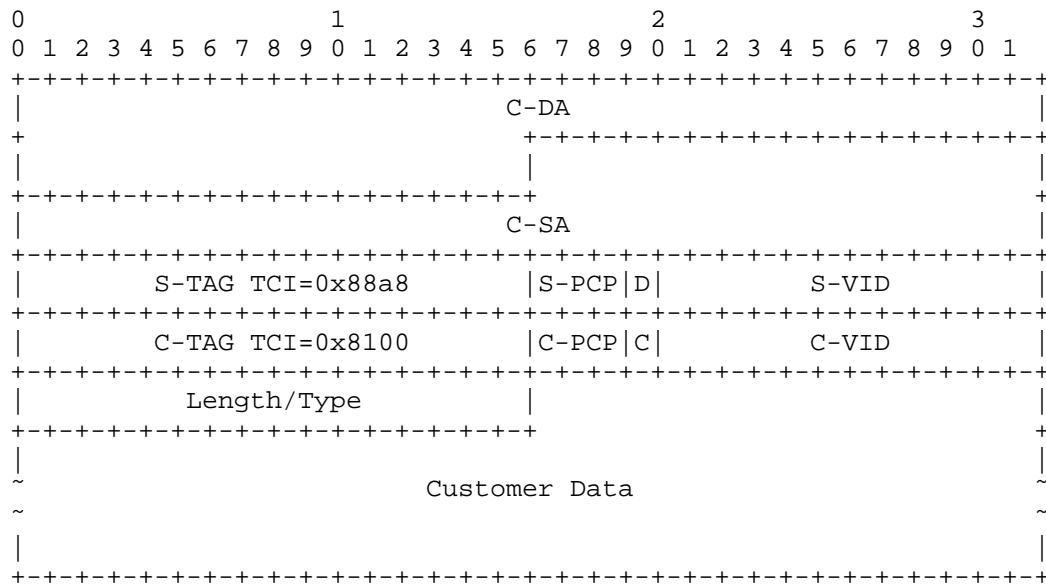


Figure B-2: IEEE802.1Qbh (S-TAG+C-TAG) Frame Format in Data Center Network

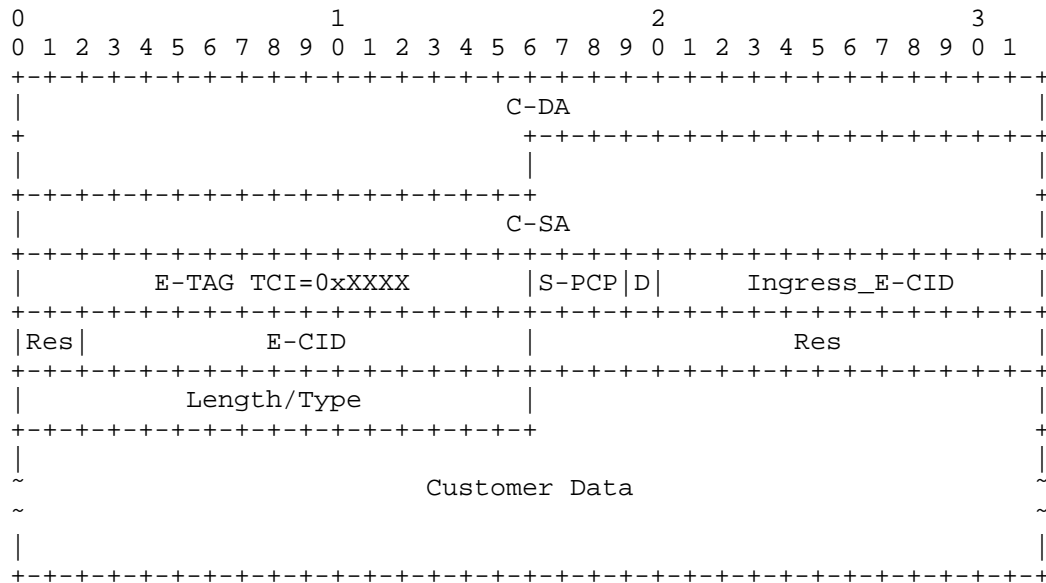


Figure B-3: IEEE802.1Qbh (E-TAG) Frame Format in Data Center Network

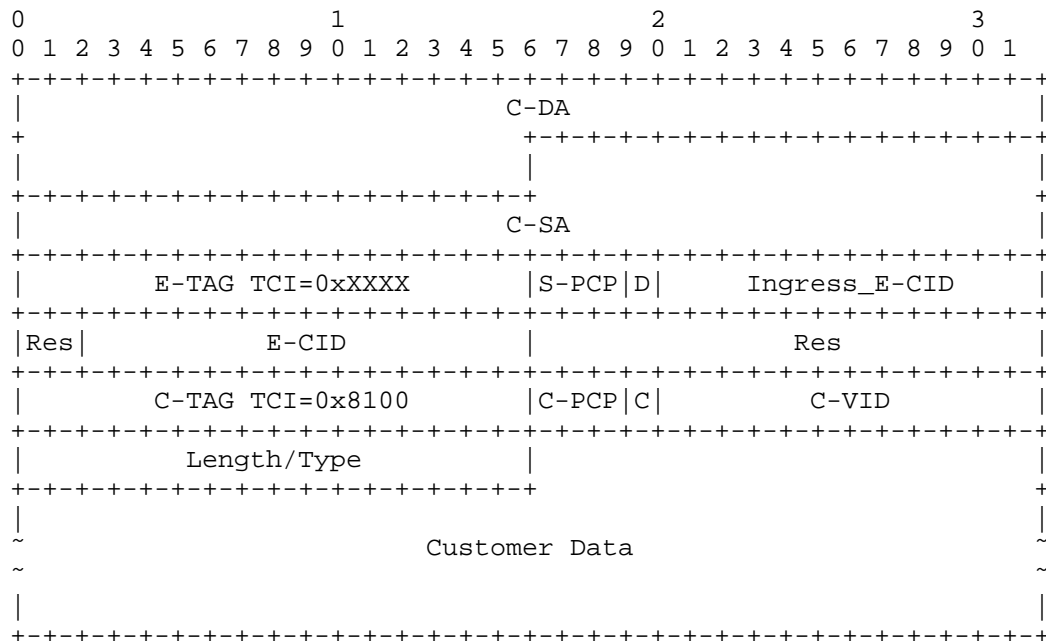


Figure B-4: IEEE802.1Qbh (E-TAG+C-TAG) Frame Format in Data Center

Network

Appendix C. Frame Format in TRILL

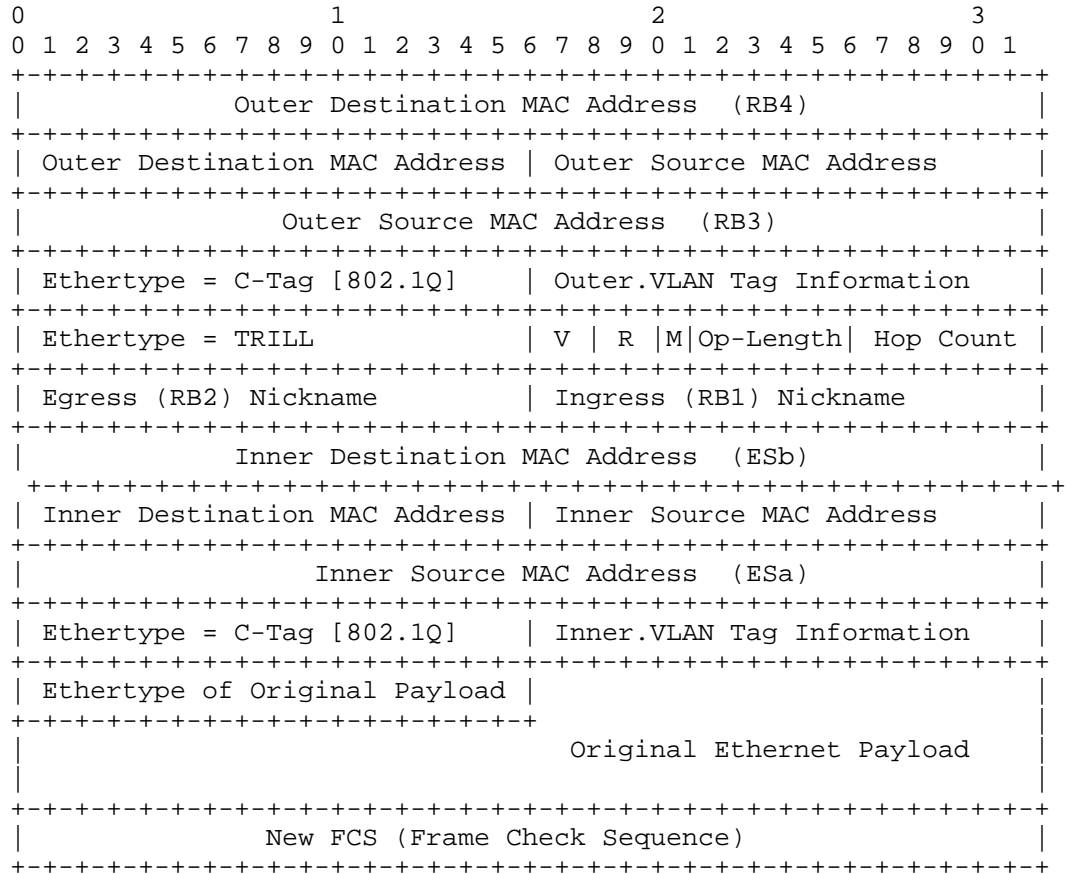


Figure C-1: TRILL Data Encapsulation over Ethernet

Appendix D. Template Formats Example

										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								
+-----																																							

Figure D-1: Template Format Example

Authors' Addresses

Shingo Kashima
 NTT Information Sharing Platform Lab.
 Midori-Cho 3-9-11
 Musashino-shi, Tokyo 180-8585
 Japan

Phone: +81 422 59 3894
 Email: kashima@nttv6.net

Kensuke Nakata
NTT Information Sharing Platform Lab.
Midori-Cho 3-9-11
Musashino-shi, Tokyo 180-8585
Japan

Phone: +81 422 59 6958
Email: kashima_shingo@mail.goo.ne.jp

Atsushi Kobayashi
NTT EAST IT Innovation Department
25F 3-20-2 Nishi-shinjuku
Shinjuku-ku, Tokyo 163-1425
Japan

Phone: +81-3-5353-3636
Email: akoba@nttv6.net

IP Flow Information Export WG
Internet-Draft
Intended status: Informational
Expires: September 15, 2011

D. Mentz
G. Muenz
L. Braun
TU Muenchen
March 14, 2011

Recommendations for Implementing IPFIX over DTLS
<draft-mentz-ipfix-dtls-recommendations-02>

Abstract

This document discusses problems and solutions regarding the implementation of the IPFIX protocol over DTLS. It updates the "IPFIX Implementation Guidelines" [RFC5153].

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on September 15, 2011.

Copyright Notice

Copyright (c) 2011 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Terminology	3
3. Issues and Recommendations Regarding IPFIX over DTLS/UDP . . .	4
3.1. Undetected Collector Crashes	4
3.1.1. Problem Description	4
3.1.2. Recommendation	5
3.1.3. Alternative Workarounds	6
3.2. Incorrect Path MTU Values	7
3.2.1. Problem Description	7
3.2.2. Recommendation	8
4. Issues and Recommendations Regarding IPFIX over DTLS/SCTP . .	9
4.1. SCTP-AUTH	9
4.2. Renegotiation for DTLS and SCTP-AUTH	9
4.2.1. Problem Description	9
4.2.2. Recommendation	10
5. Mutual Authentication via Pre-Shared Keys	10
6. Security Considerations	11
Appendix A. Acknowledgements	11
7. References	11
7.1. Normative References	11
7.2. Informative References	11
Authors' Addresses	12

1. Introduction

All implementations of the IPFIX protocol conforming to [RFC5101] must support DTLS [RFC4347] if SCTP or UDP is selected as IPFIX transport protocol. This document discusses specific issues that have arisen during the implementation of the IPFIX protocol over DTLS (the source code of the implementation is available as part of VERMONT [VERMONT]).

Section 3 discusses two issues which may lead to the loss of IPFIX Messages if DTLS is used with UDP as transport protocol: unexpected Collector crashes and wrong path MTU values. In the first case, the data loss may even not be recognized by the Collector. By following the recommendations of this document, these two problems can be avoided.

Section 4 discusses one issue which corresponds to the implementation of IPFIX over DTLS/SCTP. In this case, DTLS renegotiations require the interruption of the data export for a short period of time, which may lead to the queuing and potential loss of IPFIX Messages at the Exporting Process. For Exporters that operate at a high data rate, it is recommended to switch over to a newly established DTLS/SCTP Transport Session instead of triggering DTLS renegotiation for an existing Transport Session.

When the "IPFIX Implementation Guidelines" were published [RFC5153], no implementation of IPFIX over DTLS/UDP or DTLS/SCTP actually existed. Therefore, Sections 8.4 and 8.5 of [RFC5153] are incomplete and do not cover the issues described in this document. Hence, the recommendations of this document complement and update the "IPFIX Implementation Guidelines" [RFC5153].

Finally, Section 5 suggests to support the pre-shared key ciphersuites for TLS for mutual authentication. These ciphersuites can do without a public-key infrastructure (PKI) and can therefore facilitate the setup of an environment with a limited number of IPFIX devices.

2. Terminology

This document adopts the IPFIX terminology used in [RFC5101]. As in all IPFIX documents, all IPFIX specific terms have the first letter of a word capitalized when used in this document.

3. Issues and Recommendations Regarding IPFIX over DTLS/UDP

Regarding IPFIX over DTLS/UDP, [RFC5101] and [RFC5153] refer to [RFC4347] which specifies the usage of DTLS over the transport protocol UDP. [RFC5153] explains that Exporting Processes and Collecting Processes should behave as if UDP without DTLS was transport protocol.

During the implementation of IPFIX over DTLS/UDP, it turned out that the specification of [RFC4347] is insufficient for IPFIX data export because the loss of DTLS state at the Collecting Process may not be detected by the Exporting Process. As a consequence, it remains unnoticed that all further IPFIX Messages arriving at the Collecting Process must be discarded. This issue as well as recommendations how to solve it are discussed in Section 3.1.

For IPFIX export over UDP, [RFC5101] specifies that the total packet size of IPFIX Messages must not exceed the path MTU (PMTU). Section 8.4 of [RFC5153] points out that DTLS introduces overhead which affects the packet size. In fact, the utilization of DTLS affects the packet size, yet it does not generally result in larger packet sizes. In particular, if the IPFIX Message is compressed before being encrypted, the size of the DTLS record is likely to be smaller than the original IPFIX Message. However, since the compression ratio cannot be predicted, it is save to make conservative assumptions about the DTLS record size.

Another general problem regarding the utilization of UDP as transport protocol is that the total packet size should not exceed 512 octets if the PMTU is not available [RFC5101]. Since the PMTU is usually larger than 512 octets, this limitation causes overhead due to unnecessarily small IPFIX Messages. Hence, there is an interest to provide the Exporting Process with a correct PMTU value.

If the PMTU is known, it can be configured by the user. Otherwise, the PMTU can be determined by PMTU discovery mechanisms defined in [RFC1191] and [RFC1981]. However, these mechanisms do not always provide reliable results. Section 3.2 discusses this issue in more detail and presents a better PMTU discovery mechanism for DTLS/UDP.

3.1. Undetected Collector Crashes

3.1.1. Problem Description

DTLS has been conceived for deployment on top of unreliable transport protocols, such as UDP. Hence, the handshaking protocol of DTLS is able to cope with lost datagrams and datagrams that arrive out of order at the receiver. In contrast to UDP, which does not maintain

any connection state, DTLS has to maintain state across multiple datagrams at both endpoints. This state is established and initialized during the DTLS handshake [RFC4347].

During the DTLS handshake, the two peers authenticate each other and agree upon several parameters which are necessary to communicate over DTLS. Among these parameters are a cipher suite as well as a shared key that is usually established using a Diffie-Hellman key exchange. If one of the peers crashes unexpectedly, these parameters as well as the maintained DTLS state usually get lost. As a consequence, the peer is not able to check the integrity of newly arrived datagrams or to decrypt the datagrams' payload.

In the case of connection-oriented transport protocols, such as TCP or SCTP, a connection endpoint will be informed about the crash of its correspondent by the transport protocol. UDP, however, is connection-less, which means that the crash of the receiver is not noticed by the sender. There are situations in which the sender might receive ICMP messages indicating that the receiver is experiencing problems, for example if an ICMP port unreachable message is returned because the UDP port is closed. However, there is no guarantee that these ICMP messages will be sent. Also, implementations should ignore these messages as they are not authenticated and might therefore be forged. DTLS as specified in [RFC4347] does not provide any mechanisms for dead peer detection, thus the crash of one of the peers has to be detected and handled by protocols in the upper layers.

As IPFIX is a unidirectional protocol, a conforming implementation of an IPFIX Exporter only sends but does not receive any data. Hence, the Exporter cannot tell from the absence of returning traffic that the Collector has crashed. Instead, the Exporter keeps on sending data which must be discarded by the recovered Collector because the information needed to check the integrity and to decrypt the data is lost.

3.1.2. Recommendation

The DTLS heartbeat extension which has been suggested in [I-D.seggelmann-tls-dtls-heartbeat] allows a DTLS endpoint to detect a dead peer. With this extension, each endpoint may transmit DTLS heartbeat request messages to the other peer. Each peer is supposed to send back a heartbeat response message for every heartbeat request message it receives. As UDP provides unreliable transport, it may happen that heartbeat request or response messages are lost. Nevertheless, a peer can be declared dead if it fails to respond to a certain number of consecutive heartbeat requests.

The computational and bandwidth overhead of the heartbeat messages is very small. As another advantage, the exchange of heartbeat messages does not affect the transport of user data. In particular, the transport of user data does not have to be interrupted.

IPFIX Exporters and Collectors should support the DTLS heartbeat extension to allow an Exporting Process to check whether the Collecting Process is still able to decrypt the exported IPFIX Messages. To detect a crashed Collector, the Exporting Process must actively trigger the sending of a DTLS heartbeat request message. This should be done on a regular basis (e.g., periodically). It must be noted that a dead peer remains undetected in the time interval between two successive heartbeat requests.

The only problem with this solution is that the DTLS heartbeat extension has not yet been standardized.

3.1.3. Alternative Workarounds

If the DTLS heartbeat extension is not available, there exist two workarounds which also enable the detection of a crashed Collector. However, these approaches have several disadvantages compared to heartbeat messages.

1. The first option is to let the Exporting Process periodically trigger renegotiations on the DTLS layer. During a renegotiation, the Collecting Process has to participate in a new handshake, implying the exchange of datagrams in both direction. If a Collector has crashed, it cannot respond to the handshake messages. Thus, the absence of any return messages during the renegotiation tells the Exporter that the Collector has probably lost the DTLS state.

Under normal conditions, renegotiations are used to renew the keying material in a long living connection. Depending on whether a full or abbreviated handshake is carried out, a renegotiation can be very costly in terms of computational overhead because it involves public key operations. In addition, the DTLS specification [RFC4347] leaves open if user data can be sent while the rehandshake is in progress or if data transmission has to pause. Typical implementations, such as OpenSSL [OpenSSL], require data transmission to pause until the handshake is completed. Consequently, the export of IPFIX Messages must be stalled for at least two round trip times, which could lead to IPFIX Messages queuing up in the buffer of the Exporting Process and potential loss of data.

To make sure that the Exporter learns quickly about a crashed

Collector, renegotiations would have to be carried out on a regular basis.

2. Another approach is to periodically establish new DTLS connections and replace the existing DTLS connection by a new one. Establishing a new DTLS connection involves a bidirectional handshake which requires both peers to be alive. Two successive connections should overlap in a way such that no IPFIX Message is lost. This can be achieved by switching to the new connections only after all Templates have been sent.

This solution has the same computational overhead as the first workaround. Every DTLS connection setup might involve costly public key operations and a small overhead in terms of the transmitted packets. However, public key operations do not have to be carried out if both DTLS implementations support a feature called session resumption which allows the reuse of keying material from an earlier session.

The main advantage over periodical DTLS renegotiations is that this solution does not require to stall the transmission of user data. IPFIX records can be transmitted without interruption thanks to the overlap of the old and the new DTLS connection.

From the point of view of IPFIX, every new DTLS connection represents a new Transport Session. At the Collector side, however, the different Transport Sessions can be easily associated to the same Exporter since the Exporter IP address remains the same. At the beginning of every new Transport Session, not only all active Templates have to be sent, but also certain Data Records defined by Option Templates. In the case of UDP, however, this does not cause significant additional overhead because Templates and Data Records defined by Option Templates need to be resent periodically anyway.

3.2. Incorrect Path MTU Values

3.2.1. Problem Description

[RFC5101] states that the Exporter must not generate IPFIX Messages that result in IP packets which are larger than the PMTU. The mechanism that is commonly used to discover the PMTU is described in [RFC1191] and [RFC1981] and works as follows: The sender sets the Don't Fragment (DF) bit on all outgoing IP packets, which bans the routers on the path from fragmenting these IP packets. If a router on the path cannot forward a packet because it is larger than the MTU of the outbound link, it discards the packet and sends back an ICMP "fragmentation needed and DF set" message [RFC0792]. This message

also includes a hint about the MTU of the outbound link. Upon receiving this ICMP message, the sender updates its PMTU estimate for this specific destination IP address. In order to avoid that future packets are discarded, the sender limits, from now on, the size of IP packets to the current PMTU estimate. This new estimate may or may not be the final PMTU estimate as there are potentially other links further down the path with even smaller MTUs. The PMTU discovery process is therefore repeated until all IP packets that are as big as the PMTU estimate are delivered to the destination.

An important characteristic of this mechanism is that at least one UDP datagram is lost per update of the PMTU estimate. Hence, if deployed by an IPFIX Exporting Process, a certain number of IPFIX Messages will be lost until the final PMTU estimate is found. A more severe problem is that ICMP messages may be blocked by firewalls. As a result, the PMTU discovery mechanism fails without being noticed by the Exporting Process. Instead, the Exporting Process sticks to an incorrect PMTU estimate which is larger than the true PMTU. As a consequence, all packets which exceed the actual PMTU will be discarded on their way to the Collector, given that the "don't fragment" bit is set for all packets.

3.2.2. Recommendation

If DTLS is used, the PMTU can be determined with the DTLS heartbeat extension [I-D.seggelmann-tls-dtls-heartbeat] which has already been presented as solution to the dead peer detection problem in Section 3.1.2. This DTLS extension enables the Exporting Process to send heartbeat request messages which have the size of the PMTU estimate. If the Collecting Process acknowledges the reception of such a heartbeat request messages with a heartbeat response message, the Exporting Process knows that the PMTU estimate is less than or equal to the real PMTU to the Collector. If there is no response, the Exporting Process reduces the PMTU estimate and tries to send another heartbeat request message with the size of the new PMTU estimate. This procedure is repeated until the Exporting Process receives a heartbeat response messages. Since packets may be lost due to other reasons as well, every PMTU estimate should be probed in multiple attempts.

The described PMTU discovery mechanism can be used in conjunction with [RFC1191]. If a heartbeat request messages triggers an ICMP "fragmentation needed and DF set" message, the Exporting Process may decrease the PMTU estimate according to the returned MTU value. As a general advantage, only DTLS heartbeat messages are involved in the PMTU discovery. Hence, if the PMTU discovery using heartbeat messages is completed before starting the IPFIX export, no IPFIX Messages will be lost because of their size.

Since the PMTU may change over time due to routing changes, PMTU discovery with heartbeat messages should be repeated on a regular basis in order to ensure that the PMTU estimate is kept up to date.

4. Issues and Recommendations Regarding IPFIX over DTLS/SCTP

When [RFC5153] was published, the standardization of DTLS for SCTP was not yet completed. Therefore, the guidelines regarding the implementation of IPFIX over DTLS/SCTP are incomplete as well. In particular, [RFC5153] does not mention that DTLS for SCTP, as specified in [RFC6083], requires that the SCTP implementation supports the SCTP-AUTH extension [RFC4895]. The relationship between SCTP-AUTH and DTLS is explained in Section 4.1. As another change to [RFC5153], an implementation of DTLS for SCTP is now available at <http://sctp.fh-muenster.de/>.

If IPFIX data is exported over DTLS/SCTP, the export needs to be interrupted during DTLS renegotiations. For situations where this is unacceptable, Section 4.2 presents a workaround.

4.1. SCTP-AUTH

DTLS only protects the user data transported by SCTP. SCTP-AUTH is needed to protect SCTP control information which could otherwise be tampered with by an attacker. For example, a man-in-the-middle attacker could easily tamper with the stream ID or the payload protocol identifier of a data chunk. If PR-SCTP is used, an attacker may even suppress data chunks without being detected by forging SACK and FORWARD-TSN chunks.

SCTP-AUTH [RFC4895] deploys authentication chunks to authenticate certain types of subsequent chunks in the same packet using a hashed message authentication code (HMAC). While SCTP-AUTH enables the negotiation of the hash algorithm, it provides no means for secure key agreement. Therefore, a cross layer approach is used to extract keying material from the DTLS layer and use it in the SCTP layer. This approach is described in [RFC6083] and is readily available in OpenSSL.

4.2. Renegotiation for DTLS and SCTP-AUTH

4.2.1. Problem Description

A DTLS renegotiation (i.e., change of keying material) requires to interrupt the ongoing data transfer because DTLS does not guarantee the proper authentication and decryption of user messages that were secured with outdated keying material. The implementation has to

make sure that no data is in flight when the keying material is exchanged. This means that data transfer on all SCTP streams has to stop before a renegotiation can be initiated. Moreover, all data chunks in the send buffer need to be acknowledged before the renegotiation can start. In practice, the renegotiation has to wait until the SCTP sockets at both endpoints return SCTP_SENDER_DRY_EVENT [I-D.ietf-tsvwg-sctpsocket]. Only after the handshake has been completed, the data transfer can be resumed.

In the case of IPFIX, this means that the Exporting Process has to interrupt the export of IPFIX Messages for a certain period of time. IPFIX Messages generated in the meantime have to be buffered or dropped until the renegotiation is completed.

4.2.2. Recommendation

If an Exporting Process exports IPFIX Messages at a very high rate, it is probably impossible to buffer IPFIX Messages during a DTLS renegotiation. In order to avoid that IPFIX Messages need to be dropped at the Exporter, DTLS renegotiations should not be performed in such situations. If the keying material needs to be changed, a better solution is to establish a new DTLS/SCTP association to the same Collector. After completing the handshakes of SCTP and DTLS and after sending the IPFIX Templates on the new association, the Exporting Process switches to the new Transport Session.

Compared to a renegotiation, some overhead is produced because Templates as well as certain Data Records defined by Option Template have to be resent, which would not be necessary if the old Transport Session was kept. However, the amount of additional data that has to be sent is assumed to be rather small.

5. Mutual Authentication via Pre-Shared Keys

[RFC5101] mandates strong mutual authentication of Exporters and Collectors via asymmetric keys which are stored in X.509 certificates. This enables the user to take advantage of a public-key infrastructure (PKI) and let the endpoints verify the identity of their peers by using this infrastructure.

While a PKI is beneficial in an environment with a large number of endpoints that potentially communicate with each other, the cost of maintaining a PKI maybe disproportionate in smaller environments. [RFC4279] defines a set of new ciphersuites that use pre-shared keys instead of asymmetric keys for mutual authentication and therefore do not require a PKI. Allowing IPFIX implementations to use these ciphersuites can lower the administrative burden of setting up an

IPFIX connection that is based on DTLS or TLS. These ciphersuites are also of benefit to performance-constrained environments as they do not require computational expensive public key operations.

If the IPFIX specification allows these new ciphersuites to be used, it still has to be decided which identity type Exporters send with the ClientKeyExchange message. Refer to Section 5 of [RFC4279] for more details. The authors recommend to use the Fully Qualified Domain Name (FQDN) of the Exporter as the identity when initiating a connection. The security considerations outlined in Section 7 of [RFC4279] apply.

6. Security Considerations

The recommendations in this document do not introduce any additional security issues to those already mentioned in [RFC5101] and [RFC4279].

Appendix A. Acknowledgements

The authors thank Michael Tuexen and Robin Seggelmann for their contribution on the standardization and implementation of DTLS for SCTP as well as for their valuable advice regarding the implementation of IPFIX over DTLS.

7. References

7.1. Normative References

- [RFC4347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security", RFC 4347, April 2006.
- [RFC4279] Eronen, P. and H. Tschofenig, "Pre-Shared Key Ciphersuites for Transport Layer Security (TLS)", RFC 4279, December 2005.
- [RFC5101] Claise, B., "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information", RFC 5101, January 2008.

7.2. Informative References

- [RFC0792] Postel, J., "Internet Control Message Protocol", STD 5, RFC 792, September 1981.

- [RFC1191] Mogul, J. and S. Deering, "Path MTU discovery", RFC 1191, November 1990.
- [RFC1981] McCann, J., Deering, S., and J. Mogul, "Path MTU Discovery for IP version 6", RFC 1981, August 1996.
- [RFC3758] Stewart, R., Ramalho, M., Xie, Q., Tuexen, M., and P. Conrad, "Stream Control Transmission Protocol (SCTP) Partial Reliability Extension", RFC 3758, May 2004.
- [RFC4895] Tuexen, M., Stewart, R., Lei, P., and E. Rescorla, "Authenticated Chunks for the Stream Control Transmission Protocol (SCTP)", RFC 4895, August 2007.
- [RFC4960] Stewart, R., "Stream Control Transmission Protocol", RFC 4960, September 2007.
- [RFC5153] Boschi, E., Mark, L., Quittek, J., Stiernerling, M., and P. Aitken, "IP Flow Information Export (IPFIX) Implementation Guidelines", RFC 5153, April 2008.
- [RFC6083] Tuexen, M., Seggelmann, R., and E. Rescorla, "Datagram Transport Layer Security (DTLS) for Stream Control Transmission Protocol (SCTP)", RFC 6083, January 2011.
- [VERMONT] "VERMONT (VERsatile MONitoring Toolkit)", Homepage <http://vermont.berlios.de/>, 2010.
- [OpenSSL] "OpenSSL Cryptography and SSL/TLS Toolkit", Homepage <http://www.openssl.org/>, 2010.
- [I-D.seggelmann-tls-dtls-heartbeat] Seggelmann, R., Tuexen, M., and M. Williams, "Transport Layer Security and Datagram Transport Layer Security Heartbeat Extension", draft-seggelmann-tls-dtls-heartbeat-02 (work in progress), February 2010.
- [I-D.ietf-tsvwg-sctpsocket] Stewart, R., Tuexen, M., Poon, K., Lei, P., and V. Yasevich, "Sockets API Extensions for Stream Control Transmission Protocol (SCTP)", draft-ietf-tsvwg-sctpsocket-27 (work in progress), March 2011.

Authors' Addresses

Daniel Mentz
Technische Universitaet Muenchen
Department of Informatics
Chair for Network Architectures and Services (I8)
Boltzmannstr. 3
Garching D-85748
DE

Email: mentz@in.tum.de

Gerhard Muenz
Technische Universitaet Muenchen
Department of Informatics
Chair for Network Architectures and Services (I8)
Boltzmannstr. 3
Garching D-85748
DE

Phone: +49 89 289-18008
Email: muenz@net.in.tum.de
URI: <http://www.net.in.tum.de/~muenz>

Lothar Braun
Technische Universitaet Muenchen
Department of Informatics
Chair for Network Architectures and Services (I8)
Boltzmannstr. 3
Garching D-85748
DE

Phone: +49 89 289-18010
Email: braun@net.in.tum.de
URI: <http://www.net.in.tum.de/~braun>

IPFIX Working Group
Internet-Draft
Intended status: BCP
Expires: December 26, 2011

B. Trammell
ETH Zurich
B. Claise
Cisco Systems, Inc.
June 24, 2011

Guidelines for Authors and Reviewers of IPFIX Information Elements
draft-trammell-ipfix-ie-doctors-02.txt

Abstract

This document provides guidelines for the definition of IPFIX Information Elements for addition to the IANA IPFIX Information Element registry, in order to extend the applicability of the IPFIX protocol to new operations and management areas.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on December 26, 2011.

Copyright Notice

Copyright (c) 2011 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
1.1. Intended Audience and Usage	3
1.2. Overview of relevant IPFIX documents	4
2. Terminology	4
3. How to apply IPFIX	5
4. Defining new Information Elements	6
4.1. Information Element naming	6
4.2. Information Element data types	7
4.3. Information Element numbering	7
4.4. Ancillary Information Element properties	8
4.5. Internal structure in Information Elements	8
4.6. Enumerated Values and Subregistries	9
4.7. Reversibility as per RFC 5103	10
4.8. Promotion of Enterprise-Specific Information Elements	10
5. The Information Element Lifecycle	10
5.1. The IE-DOCTORS process	11
5.2. Revising Information Elements	11
5.3. Deprecating Information Elements	12
5.4. Versioning the entire IANA Registry	13
6. When not to define new Information Elements	14
6.1. Maximizing reuse of existing Information Elements	14
6.2. Applying enterprise-specific Information Elements	15
7. Applying IPFIX to non-Flow Applications	16
8. Writing Internet-Drafts for IPFIX Applications	16
8.1. Example Information Element Definition	17
8.2. Defining Recommended Templates	17
9. A Textual Format for Specifying Information Elements and Templates	18
9.1. Information Element Specifiers	19
9.2. Specifying Templates	21
9.3. Specifying IPFIX Structured Data	21
10. Security Considerations	22
11. IANA Considerations	23
12. Acknowledgements	23
13. References	23
13.1. Normative References	23
13.2. Informative References	24
Authors' Addresses	25

1. Introduction

This document provides guidelines for the extension of the applicability of the IP Flow Information Export (IPFIX) protocol to network operations and management purposes outside the initial scope defined in "IPFIX Applicability Statement" [RFC5472]. These new applications are largely defined through the definition of new Information Elements beyond those defined in the IPFIX Information Model [RFC5102] or already added to the IANA IPFIX Information Element Registry [iana-ipfix-assignments]. New applications may be further specified through additional RFCs defining and describing their usage.

We intend this document to enable the expansion of the applicability of IPFIX to new areas by experts in the working group or area directorate concerned with the technical details of the protocol or application to be measured or managed using IPFIX. This expansion would occur with the consultation of IPFIX experts informally called 'IE-Doctors'. It provides guidelines both for those defining new Information Elements as well as the IE-Doctors reviewing them.

1.1. Intended Audience and Usage

This document is meant for two separate audiences. For IETF contributors extending the applicability of IPFIX, it provides a set of guidelines and best practices to be used in deciding which Information Elements are necessary for a given existing or new application, defining these Information Elements, and deciding whether an RFC should be published to further describe the application. For the IPFIX experts appointed as IE-Doctors, and for IANA personnel changing the Information Element registry, it defines a set of acceptance criteria against which these proposed Information Elements should be evaluated.

This document is not intended to guide the extension of the IPFIX protocol itself, e.g. through new export mechanisms, data types, or the like; these activities should be pursued through the publication of standards-track RFCs by the IPFIX Working Group.

This document specifies additional practices beyond those appearing in the IANA Considerations sections of existing IPFIX documents, especially the Information Model [RFC5102]. The practices outlined in this document are intended to guide experts when making changes to the IANA registry under Expert Review as defined in [RFC5226].

1.2. Overview of relevant IPFIX documents

[RFC5101] defines the IPFIX Protocol, the IPFIX-specific terminology used by this document, and the data type encodings for each of the data types supported by IPFIX.

[RFC5102] defines the initial IPFIX Information Model, as well as procedures for extending the Information Model. It states that new Information Elements may be added to the Information Model on Expert Review basis, and delegates the appointment of experts to an IESG Area Director. This document is intended to further codify the best practices to be followed by these experts, in order to improve the efficiency of this process.

[RFC5103] defines a method for exporting bidirectional flow information using IPFIX; this document should be followed when extending IPFIX to represent information about bidirectional network interactions in general. Additionally, new Information Elements should be annotated for their reversibility or lack thereof as per this document.

[RFC5610] defines a method for exporting information about Information Elements inline within IPFIX. In doing so, it explicitly defines a set of restrictions on the use of data types and semantics which are implied in [RFC5101] and [RFC5102]; these restrictions **MUST** be observed in the definition of new Information Elements, as in Section 4.4.

2. Terminology

Capitalized terms used in this document that are defined in the Terminology section of [RFC5101] are to be interpreted as defined there.

An "application", as used in this document, refers to a candidate protocol, task, or domain to which IPFIX export, collection, and/or storage is applied, beyond those within the IPFIX Applicability statement [RFC5472]. By this definition, PSAMP [RFC5476] was the first new IPFIX application after the publication of the IPFIX protocol [RFC5101].

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].

3. How to apply IPFIX

Though originally specified for the export of IP flow information, the message format, template mechanism, and data model specified by IPFIX lead to it being applicable to a wide variety of network management situations. In addition to flow information export, for which it was designed, and packet information export as specified by PSAMP [RFC5476], any application with the following characteristics is a good candidate for an IPFIX application:

- o The application's data flow is fundamentally unidirectional. IPFIX is a "push" protocol, supporting only the export of information from a sender (an Exporting Process) to a receiver (a Collecting Process). Request-response interactions are not supported by IPFIX.
- o The application handles discrete event information, or information to be periodically reported. IPFIX is particularly well suited to representing events, which can be scoped in time.
- o The application handles information about network entities. IPFIX's information model is network-oriented, so network management applications have many opportunities for information model reuse.
- o The application requires a small number of arrangements of data structures relative to the number of records it handles. The template-driven self-description mechanism used by IPFIX excels at handling large volumes of identically structured data, compared to representations which define structure inline with data (such as XML).

Most applications meeting these criteria can be supported over IPFIX. Once it's been determined that IPFIX is a good fit, the next step is determining which Information Elements are necessary to represent the information required by the application. Especially for network-centric applications, the IPFIX Information Element registry may already contain all the necessary Information Elements (see Section 6.1 for guidelines on maximizing Information Element reuse). In this case, no additional work within the IETF is necessary: simply define Templates and start exporting.

It is expected, however, that most applications will be able to reuse some existing Information Elements, but must define some additional Information Elements to support all their requirements; in this case, see Section 4 for best practices to be followed in defining Information Elements.

Optionally, a Working Group or individual contributor may choose to publish an RFC detailing the new IPFIX application. Such an RFC should contain discussion of the new application, the Information Element definitions as in Section 4, as well as suggested Templates and examples of the use of those Templates within the new application as in Section 8.2. Section 9 defines a compact textual Information Element notation to be used in describing these suggested Templates and/or the use of IPFIX Structured Data [I-D.ietf-ipfix-structured-data] within the new application.

4. Defining new Information Elements

In many cases, a new application will require nothing more than a new Information Element or set of Information Elements to be exportable using IPFIX. An Information Element meeting the following criteria, as evaluated by appointed IPFIX experts, is eligible for inclusion in the Information Element registry:

- o The Information Element MUST be sufficiently unique within the registry. A proposed Information Elements which is a substantial duplicate of an exiting Information Element is to be represented using the existing Element.
- o The Information Element SHOULD contain minimal internal structure; complex information should be represented with multiple simple Information Elements to be exported in parallel, as in Section 4.5.
- o The Information Element SHOULD be generally applicable to the application at hand, which SHOULD be of general interest to the community. Information Elements representing information about proprietary or nonstandard applications SHOULD be represented using enterprise-specific Information Elements as detailed in section 6.2 of [RFC5101].

The definition of new Information Elements requires a descriptive name, a specification of the data type as one from the IPFIX Data Type Registry, and a human-readable description written in English. This section provides guidelines on each of these components of an Information Element definition, referring to existing documentation such as [RFC5102] as appropriate.

4.1. Information Element naming

Information Element Names should be defined in accordance with section 2.3 of [RFC5102]; the most important naming conventions are repeated here for convenience.

- o Names of Information Elements should be descriptive.
- o Names of Information Elements MUST be unique within the IPFIX information model.
- o Names of Information Elements start with non-capitalized letters.
- o Composed names use capital letters for the first letter of each component (except for the first one). All other letters are non-capitalized, even for acronyms. Exceptions are made for acronyms containing non-capitalized letter, such as 'IPv4' and 'IPv6'. Examples are sourceMacAddress and destinationIPv4Address.

In addition, new Information Elements pertaining to a specific protocol SHOULD name the protocol in the first word in order to ease searching by name (e.g. "sipMethod" for a SIP method, as would be used in a logging format for SIP based on IPFIX). Similarly, new Information Elements pertaining to a specific application SHOULD name the application in the first word.

4.2. Information Element data types

IPFIX provides a set of data types covering most primitives used in network measurement and management applications. The most appropriate data type should be chosen for the Information Element type, out of the IPFIX informationElementDataTypes subregistry at [iana-ipfix-assignments].

Because IPFIX provides reduced-length encoding for Information Elements, unless an integral Information Element is derived from a fixed-width field in a measured protocol (e.g., tcpSequenceNumber, which is an unsigned32), it should be defined with the maximum possible width, generally signed64 or unsigned64. Applications can then choose to use reduced-size encoding as defined in Section 6.2 of [RFC5101] in cases where fewer than 2^{64} values are necessary.

Information Elements representing time values should be exported with appropriate precision. For example, a Information Element for a time measured at second-level precision should be defined as having a dateTimeSeconds data type, instead of dateTimeMilliseconds.

4.3. Information Element numbering

In general, when adding newly registered Information Elements to the registry, IANA SHOULD assign the lowest available Information Element identifier (the value column in [iana-ipfix-assignments] in the range 128-32767, noting that prior noncontiguous allocation may lead to unassigned Information Elements with lower Information Element

identifiers than some presently assigned Information Elements. This is the case with the PSAMP Information Model [RFC5477], which assigned a block of Information Elements identifiers starting at 300.

Information Element identifiers in the range 1-128 MUST NOT be assigned unless the Information Element is compatible with the NetFlow v9 protocol as described in [RFC3954]. Such Information Elements may ONLY be requested by a NetFlow v9 expert, to be designated by the IESG to consult with IANA on NetFlow v9 compatibility within IPFIX.

4.4. Ancillary Information Element properties

Information Elements to which special semantics apply SHOULD define these semantics with one of the values in the Information Element Semantics registry, as described in Section 3.2 of [RFC5102], subject to the restrictions given in Section 3.10 of [RFC5610]; essentially, the semantics and the type must be consistent.

When defining Information Elements representing a dimensioned quantity or entity count, the units of that quantity SHOULD be defined in the units field. This field takes its values from the IANA Information Element Units registry. If an Information Element expresses a quantity in units not yet in this registry, then the unit must be added to the Units registry at the same time the Information Element is added to the Information Element registry.

Additionally, when the range of values an Information Element can take is smaller than the range implied by its data type, the range SHOULD be defined within the Information Element registry.

4.5. Internal structure in Information Elements

Unless defining an Information Element which is a direct copy of a bitfield or other structured entity (e.g., the tcpControlBits Information Element for the flags byte from the TCP header) in a measured protocol, the definition of Information Elements with internal structure with the structure defined in the Description field is discouraged. In this case, the field SHOULD be decomposed into multiple primitive Information Elements to be used in parallel. For more complicated semantics, where the structure is not identical from Data Records to Data Records, use the IPFIX Structured Data [I-D.ietf-ipfix-structured-data] extension instead.

As an example of information element decomposition, consider an application-level identifier called an "endpoint", which represents a {host, port, protocol} tuple. Instead of allocating an opaque, structured "source endpoint" Information Element, the source endpoint

should be represented by three separate Information Elements: "source address", "source port", "transport protocol". In this example, the required information elements already exist in the Information Element registry: sourceIPv4Address or sourceIPv6Address, sourceTransportPort, protocolIdentifier. Indeed, as well as being good practice, this normalization down to non-structured Information Elements also increases opportunities for reuse as in Section 6.1.

The decomposition of data with internal structure SHOULD avoid the definition of Information Elements with a meaning too specific to be generally useful, or that would result in either the export of meaningless data or a multitude of templates to handle different multiplicities. A specific example of this within the IANA registry is the following list of assigned IPFIX Information Elements: mplsTopLabelStackSection, mplsLabelStackSection2, mplsLabelStackSection3, mplsLabelStackSection4, mplsLabelStackSection5, mplsLabelStackSection6, mplsLabelStackSection7, mplsLabelStackSection8, mplsLabelStackSection9, and mplsLabelStackSection10. The only distinction between those almost-identical Information Elements is the position within the MPLS stack. This Information Element design pattern met an early requirement of the definition of IPFIX which was not carried forward into the final specification -- namely, that no semantic dependency was allowed between Information Elements in the same Record -- and as such SHOULD NOT be followed in the definition of new Information Elements. In this case, since the size of the MPLS stack will vary from flow to flow, it should be exported using IPFIX Structured Data [I-D.ietf-ipfix-structured-data] where supported, as a basicList of MPLS label entries.

Note that a Template may contain multiple instances of the same Information Element; in this case, the each of the Information Elements in the Template are semantically indistinguishable, and appear in their "natural" order, where natural order is defined according to application; PSAMP uses this for exporting selectors. Multiple Information Elements used in this way are preferable to Information Elements with internal structure, but only when there is some natural order, and no semantic interdependence among the elements.

4.6. Enumerated Values and Subregistries

When defining an Information Element that takes an enumerated value from a set of values which may change in the future, this enumeration MUST be defined by an IANA registry or subregistry. For situations where an existing registry defines the enumeration (e.g., the IANA Protocol Numbers registry for the protocolIdentifier Information Element), that registry MUST be used. Otherwise, a new IPFIX

subregistry must be defined for the enumerated value, to be modified subject to Expert Review [RFC5226].

4.7. Reversibility as per RFC 5103

[RFC5103] defines a method for exporting bidirectional flows using a special Private Enterprise Number to define reverse-direction variants of IANA Information Elements, and a set of criteria for determining whether an Information Element may be reversed using this method. Section 6.1 of [RFC5103] states that Collecting Processes should use the set of criteria therein to determine reversibility. Since almost all Information Elements are reversible, these criteria are expressed as to determine the exceptions, i.e. which Information Elements are NOT reversible.

To ease the determination of reversibility, future Information Elements which are NOT reversible SHOULD note this fact in the description at the time of definition.

4.8. Promotion of Enterprise-Specific Information Elements

Some Information Elements may start their lifecycle outside the IANA registry as enterprise-specific Information Elements scoped to a Private Enterprise Number. One stated goal of enterprise-specific Information Elements is pre-standards product delivery and experimentation; should these experiments be successful and the Information Elements generally useful, these SHOULD subsequently registered with IANA.

In order to support transition from experimental registration to IANA registration, the IANA registry provides an optional "enterprise-specific IE reference" column for each Information Element. In cases of promoted enterprise-specific Information Elements, this column in the registry SHOULD contain the private enterprise and Information Element numbers of the enterprise-specific version of the Information Element.

5. The Information Element Lifecycle

Once an Information Element or set of Information Elements has been identified for a given application, Information Element specifications in accordance with Section 4 are submitted to IANA to follow the IE-DOCTORS process, as defined below. This process is also used for other changes to the registry, such as deprecation or revision, as described later in this section.

5.1. The IE-DOCTORS process

Requests to change the IANA Information Element registry or a linked subregistry are submitted to IANA, which forwards the request to a designated group of experts (IE-DOCTORS) appointed by the IETF Operations Area Directors. This group of experts reviews the request for compliance with this document, compliance with other applicable IPFIX-related RFCs, and consistency with the currently defined set of Information Elements.

IE-DOCTORS reviewers should endeavor to complete referred reviews in a timely manner. If the request is acceptable, the IE-DOCTORS signify their approval to IANA, which changes the IANA Information Element registry. If the request is not acceptable, the IE-DOCTORS can coordinate with the requestor to change the request to be compliant. The IE-DOCTORS may also choose in exceptional circumstances to reject clearly frivolous or inappropriate change requests outright.

5.2. Revising Information Elements

The Information Element status field in the Information Element Registry is defined in [RFC5102] to allow Information Elements to be 'current', 'deprecated' or 'obsolete'. No Information Elements are as of this writing deprecated or obsolete, and [RFC5102] does not define any policy for using them. Additionally, no policy is defined for revising Information Element registry entries or addressing errors therein. To be certain, changes and deprecations within the Information Element registry are not encouraged, and should be avoided to the extent possible. However, in recognition that change is inevitable, this section is intended to remedy this situation.

The primary requirement in the definition of a policy for managing changes to existing Information Elements is avoidance of interoperability problems; IPFIX experts appointed to review changes to the Information Element Registry MUST work to maintain interoperability above all else. Changes to Information Elements already in use may only be done in an interoperable way; necessary changes which cannot be done in a way to allow interoperability with unchanged implementations MUST result in deprecation.

A change to an Information Element is held to be interoperable only when:

- o it involves the correction of an error which is obviously only editorial; or

- o it corrects an ambiguity in the Information Element's definition, which itself leads to non-interoperability (e.g., a prior change to `ipv6ExtensionHeaders`); or
- o it expands the Information Element's data type without changing how it is represented (e.g., changing `unsigned32` to `unsigned64`, as with a prior change to `selectorId`); or
- o it defines a previously undefined or reserved enumerated value, or one or more previously reserved bits in an Information Element with flag semantics; or
- o it expands the set of permissible values in the Information Element's range; or
- o it harmonizes with an external reference which was itself corrected.

A non-interoperable Information Element change may also be made if it can be reasonably assumed in the eyes of the appointed experts that no unchanged implementation of the Information Element exists; this can be held to happen if a non-interoperable change to an Information Element defined shortly before is proposed to the IPFIX mailing list by the original proposer of the Information Element, and no objection is raised within a reasonable amount of time, to be defined by the expert reviewers.

If a change is permissible, it is sent to IANA, which passes it to the appointed experts for review; if there is no objection to the change from any appointed expert, IANA makes the change in the Information Element Registry. The requestor of the change is appended to the Requestor in the registry.

Each Information Element in the IANA registry has a revision number, starting at zero. Each change to an Information Element following this process increments the revision number by one. Since any revision must be interoperable according to the criteria above, there is no need for the IANA registry to store information about old revisions.

5.3. Deprecating Information Elements

Changes that are not permissible by these criteria may only be handled by deprecation. An Information Element MAY be deprecated and replaced when:

- o the Information Element definition has an error or shortcoming which cannot be permissibly changed as above; or
- o the deprecation harmonizes with an external reference which was itself deprecated through that reference's accepted deprecation method; or
- o changes in the IPFIX Protocol or its extensions, or in community understanding thereof, allow the information represented by the Information Element to be represented in a more efficient or convenient way. Deprecation in this circumstance additionally requires the assent of the IPFIX Working Group, and should be specified in the Internet Draft(s) defining the protocol change.

A request for deprecation is sent to IANA, which passes it to the IE-DOCTORS for review, as above. When deprecating an Information Element, the Information Element description **MUST** be updated to explain the deprecation, as well as to refer to any new Information Elements created to replace the deprecated Information Element. The revision number of an Information Element is incremented upon deprecation.

Deprecated Information Elements **SHOULD** continue to be supported by Collecting Processes, but **SHOULD NOT** be exported by Exporting Processes. The use of deprecated Information Elements **SHOULD** result in a log entry or human-readable warning at the Exporting and Collecting Processes. After a period of time determined in the eyes of the IE-DOCTORS experts to be reasonable in order to allow deployed Exporting Processes to be updated to account for the deprecation, a deprecated Information Element may be made obsolete. Obsolete Information Elements **MUST NOT** be supported by either Exporting or Collecting Processes. The receipt of obsolete Information Elements **SHOULD** be logged by the Collecting Process.

Names of deprecated Information Elements **MUST NOT** be reused. Names of obsolete Information Elements **MAY** be reused, but this is **NOT RECOMMENDED**, as it may cause confusion among users.

5.4. Versioning the entire IANA Registry

Consider a typical Collector implementation, which regularly downloads the entire registry in order to be compliant with the latest of set of supported IEs. While a registry revision number might seem advantageous for the Collector at first glance (avoiding the one by one comparison of all IE revisions), it is not necessary, as the IPFIX IANA registry specifies the date at which the registry was last updated in the "Last Updated" field. For purposes of identifying the latest set of Information Element versions specified

in registry, the last revision date of the Information Element registry (available in the registry XML source, or from the Last-Modified: header of [iana-ipfix-assignments]) SHOULD be used.

6. When not to define new Information Elements

Also important in defining new applications is avoiding redundancy and clutter in the Information Element registry. Here we provide guidelines for reuse of existing Information Elements, as well as guidelines on using enterprise-specific Information Elements instead of adding Information Elements in the registry.

6.1. Maximizing reuse of existing Information Elements

Whenever possible, new applications should prefer usage of existing IPFIX Information Elements to the creation of new Information Elements. IPFIX already provides Information Elements for every common Layer 4 and Layer 3 packet header field in the IETF protocol suite, basic Layer 2 information, basic counters, timestamps and time ranges, and so on. When defining a new Information Element similar to an existing one, reviewers shall ensure that the existing one is not applicable.

Note that this guideline to maximize reuse does not imply that an Information Element that represents the same information from a packet as a existing Information Element should not be added to the registry. For example, consider the `ipClassOfService` (Element ID 5), `ipDiffServCodePoint` (Element ID 98), and `ipPrecedence` (Element ID 196) Information Elements. These all represent subsets of the same field in an IP version 4 packet header, but different uses of these bits. The representation in one or another of these Information Elements contains information in itself as to how the bits were interpreted by the Metering Process.

On the other hand, simply changing the context in which an Information Element will be used is insufficient reason for the definition of a new Information Element. For example, an extension of IPFIX to log detailed information about HTTP transactions alongside network-level information should not define `httpClientAddress` and `httpServerAddress` Information Elements, preferring instead the use of `sourceIPv[46]Address` and `destinationIPv[46]Address`.

Applications dealing with bidirectional interactions should use Bidirectional Flow Support for IPFIX [RFC5103] to represent these interactions.

Specifically, existing timestamp and time range Information Elements should be reused for any situation requiring simple time stamping of an event: for single observations, the observationTime* Information Elements from PSAMP are provided, and for events with a duration, the flowStart* and flowEnd* Information Elements suffice. This arrangement allows minimal generic time handling by existing Collecting Processes and analysis workflows. New timestamp Information Elements should ONLY be defined for semantically distinct timing information (e.g., an IPFIX-exported record containing information about an event to be scheduled in the future).

In all cases the use of absolute timestamp Information Elements (e.g. flowStartMilliseconds) is RECOMMENDED, as these Information Elements allow for maximum flexibility in processing with minimal overhead. Timestamps based on the export time header in the enclosing IPFIX Message (e.g. flowStartTimeDeltaMicroseconds) MAY be used if high-precision timing is important, export bandwidth or storage space is limited, timestamps comprise a relatively large fraction of record size, and the application naturally groups records into IPFIX Messages. Timestamps based on information which must be exported in a separate Data Record defined by an Options Template (e.g. flowStartSysUpTime) MAY be used only in the context of an existing practice of using runtime-defined epochs for the given application. New applications SHOULD avoid these structures when possible.

6.2. Applying enterprise-specific Information Elements

IPFIX provides a mechanism for defining enterprise-specific Information Elements, as in Section 3.2 of [RFC5101]. These are scoped to a vendor's or organization's Structure of Management Information (SMI) Private Enterprise Number, and are under complete control of the organization assigning them.

For situations in which interoperability is unimportant, new information SHOULD be exported using enterprise-specific Information Elements instead of adding new Information Elements to the registry. These situations include:

- o export of implementation-specific information, or
- o export of information derived in a commercially-sensitive or proprietary method, or
- o export of information or meta-information specific to a commercially-sensitive or proprietary application.

While work within the IETF generally does not fall into these categories, enterprise-specific Information Elements are also useful

for pre-standardization testing of a new IPFIX application. While performing initial development and interoperability testing of a new application, the Information Elements used by the application SHOULD NOT be submitted to IANA for inclusion in the registry. Instead, these experimental Information Elements SHOULD be represented as enterprise-specific until their definitions are finalized, then transitioned from enterprise-specific to IANA-defined upon finalization. To support this transition, the IANA registry provides an experimental IE reference as defined in Section 4.8.

7. Applying IPFIX to non-Flow Applications

At the core of IPFIX is its definition of a Flow, a set of packets sharing some common properties crossing an observation point within a certain time window. However, the reliance on this definition does not preclude the application of IPFIX to domains which are not obviously handling flow data according to it. Most network management data collection tasks, those to which IPFIX is most applicable, have at their core the movement of packets from one place to another; by a liberal interpretation of the common properties defining the flow, then, almost any event handled by these can be held to concern data records conforming to the IPFIX definition of a Flow.

Non-flow information defining associations or key-value pairs, on the other hand, are defined by IPFIX Options Templates. Here, the Information Elements within an Options Template Record are divided into Scope Information Elements which define the key, and non-scope Information Elements which define the values associated with that key. Unlike Flows, Data Records defined by Options Template are not necessarily scoped in time; these Data Records are generally held to be in effect until a new set of values for a specific set of keys is exported. While this mechanism is often used by IPFIX to export metadata about the collection infrastructure, it is applicable to any association information.

An IPFIX application can mix Data Records described either type of template in an IPFIX Message or Message stream, and exploit relationships among the Flow Keys, values, and Scopes to create interrelated data structures. See [RFC5473] for an example application of this.

8. Writing Internet-Drafts for IPFIX Applications

When a new application is complex enough to require additional clarification or specification as to the use of the defined

Information Elements, this may be given in an Internet-Draft. Internet-Drafts for new IPFIX applications are best submitted to a Working Group with expertise in the area of the new application, or as independent submissions.

When defining new Information Elements in an Internet-Draft, the Internet-Draft SHOULD contain a section (or subsection) for each Information Element, which contains the attributes in Section 4 in human-readable form. An example subsection is given below. These Information Element descriptions SHOULD NOT assign Information Element numbers, instead using placeholder identifiers for these numbers (e.g. "AAA", "BBB", "CCC", or "TBD1", "TBD2", "TBD3") and a note to IANA in the IANA Considerations section to replace those placeholders in the document with Information Element numbers when the numbers are assigned. The use of these placeholder definitions allows references to the numbers in e.g. box-and-line diagrams or template definitions as in Section 9.

8.1. Example Information Element Definition

This is an example of an Information Element definition which would appear in an Internet-Draft. The name appears in the section title.

Description: Description goes here.

Data Type: Data type goes here; obligatory

Data Type Semantics: Data type semantics, if any, go here; optional

Units: Units, if any, go here; optional

Range: Range, if not implied by the data type, goes here; optional

References: References to other RFCs or documents outside the IETF, in which additional information is given, or which are referenced by the description, go here; optional

ElementId: TBD1

8.2. Defining Recommended Templates

New IPFIX applications SHOULD NOT, in the general case, define fixed templates for export, as this throws away much of the flexibility afforded by IPFIX. However, fixed template export is permissible in the case that the export implementation must operate in a resource constrained environment, and/or that the application is replacing an existing fixed-format binary export format in a maximally compatible way. In any case, Collecting Processes for such applications SHOULD

support reordered Templates or Templates with additional Information Elements.

An Internet-Draft clarifying the use of new Information Elements SHOULD include any recommended Template or Options Template Records necessary for supporting the application, as well as examples of records exported using these Template Records. In defining these Template Records, such Internet-Drafts SHOULD mention, subject to rare exceptions as above:

- o that the order of Information Elements within a Template is not significant;
- o that Templates on the wire for the application may also contain additional Information Elements beyond those specified in the recommended Template;
- o that a stream of IPFIX Messages supporting the application may also contain Data Records not described by the recommended Templates; and
- o that any reader of IPFIX Messages supporting the application MUST accept these conditions.

Definitions of recommended Template Records for flow-like information, where the Flow Key is well-defined, SHOULD indicate which of the Information Elements in the recommended Template are Flow Keys.

Recommended Templates are defined, for example, in [RFC5476] for PSAMP packet reports (section 6.4) and extended packet reports (section 6.5). Recommended Options Templates are defined extensively throughout the IPFIX documents, including in the protocol document itself [RFC5101] for exporting export statistics; in the file format [RFC5655] for exporting file metadata; and in Mediator intermediate process definitions such as [I-D.ietf-ipfix-anon] for intermediate process metadata. The discussion in these examples is a good model for recommended template definitions.

9. A Textual Format for Specifying Information Elements and Templates

The examples given above are all expressed using bitmap diagrams of the respective Templates. These are illustrative of the wire representation of simple Templates, but not particularly readable for more complicated recommended Templates, provide no support for rapid implementation of new Templates, and do not adequately convey the optional nature of ordering and additional Information Elements as

above. Therefore, we define a RECOMMENDED textual format for specifying Information Elements and Templates in Internet-Drafts in this section.

Here we define a simple textual syntax for describing IPFIX Information Elements and IPFIX Templates, with human readability, human writability, compactness, and ease of parser/generator implementation without requiring external XML support as design goals. It is intended both for use in human communication (e.g., in new Internet-Drafts containing higher-level descriptions of IPFIX Templates, or describing sets of new IPFIX Information Elements for supporting new applications of the protocol) as well as at runtime by IPFIX implementations.

9.1. Information Element Specifiers

The basis of this format is the textual Information Element Specifier, or IESpec. An IESpec contains each of the four important aspects of an Information Element: its name, its number, its type, and its size, separated by simple markup based on various types of brackets. Fully-qualified IESpecs may be used to specify existing or new Information Elements within an Information Model, while either fully-qualified or partial IESpecs may be used to define fields in a Template.

Bare words are used for Information Element names, and each aspect of information associated with an Information Element is associated with a type of brackets:

- o () parentheses for Information Element numbers,
- o < > angles for Information Element data types, and
- o [] square brackets for Information Element sizes.
- o { } curly braces contain an optional space-separated list of context identifiers to be associated with an Information Element, as described in more detail in Section 9.2

The symbol + is reserved for Information Element nesting within structured data elements; these are described in and Section 9.3, respectively.

Whitespace in IESpecs is insignificant; spaces can be added after each element in order, e.g., to align columns for better readability.

The basic form of a fully-qualified IESpec for an IANA-registered Information Element is as follows:

`name(number)<type>[size]`

where 'name' is the name of the Information Element in UTF-8, 'number' is the Information Element as a decimal integer, 'type' is the name of the data type as in the IANA `informationElementDataTypes` registry, and 'size' is the length of the Information Element in octets as a decimal integer, where 65535 or the string 'v' signifies a variable-length Information Element. [size] may be omitted; in this case, the data type's native or default size is assumed.

The basic form of a fully-qualified IESpec for an enterprise-specific Information Element is as follows:

`name(pen/number)<type>[size]`

where 'pen' is the Private Enterprise Number as a decimal integer.

A fully-qualified IESpec is intended to express enough information about an Information Element to decode and display Data Records defined by Templates containing that Information Element. Range, unit, semantic, and description information, as in [RFC5610], is not supported by this syntax.

Example fully-qualified IESpecs follow:

`octetDeltaCount(1)<unsigned64>[8]`

`octetDeltaCount(1)<unsigned64>` (unsigned64 is natively 8 octets long)

`sourceIPv4Address(8)<ipv4Address>`

`wlanSSID(146)<string>[v]`

`sipRequestURI(35566/403)<string>[65535]`

A partial IESpec is any IESpec that is not fully-qualified; these are useful when defining templates. A partial IESpec is assumed to take missing values from its canonical definition, for example, the IANA registry. At minimum, a partial IESpec must contain a name, or a number. Any name, number, or type information given with a partial IESpec must match the values given in the Information Model; however, size information in a partial IESpec overrides size information in the Information Model; in this way, IESpecs can be used to express reduced-length encoding for Information Elements.

Example partial IESpecs follow:

- o `octetDeltaCount`
- o `octetDeltaCount[4]` (reduced-length encoding)
- o (1)
- o (1)[4] (reduced length encoding; note that this is exactly equivalent to an Information Element specifier in a Template)

9.2. Specifying Templates

A Template can then be defined simply as an ordered, newline-separated sequence of IESpecs. IESpecs in example Templates illustrating a new application of IPFIX SHOULD be fully-qualified. Flow Keys may be optionally annotated by appending the {key} context to the end of each Flow Key specifier. A template counting packets and octets per five-tuple with millisecond precision in IESpec syntax is shown below.

```
flowStartMilliseconds(152)<dateTimeMilliseconds>[8]
flowEndMilliseconds(153)<dateTimeMilliseconds>[8]
octetDeltaCount(1)<unsigned64>[8]
packetDeltaCount(2)<unsigned64>[8]
sourceIPv4Address(8)<ipv4Address>[4]{key}
destinationIPv4Address(12)<ipv4Address>[4]{key}
sourceTransportPort(7)<unsigned16>[2]{key}
destinationTransportPort(11)<unsigned16>[2]{key}
protocolIdentifier(4)<unsigned8>[1]{key}
```

An Options Template is specified similarly. Scope is specified appending the {scope} context to the end of each IESpec for a Scope IE. Due to the way Information Elements are represented in Options Templates, all {scope} IESpecs must appear before any non-scope IESpec. The Flow Key Options Template defined in section 4.4 of [RFC5101] in IESpec syntax is shown below:

```
templateId(145)<unsigned16>[2]{scope}
flowKeyIndicator(173)<unsigned64>[8]
```

9.3. Specifying IPFIX Structured Data

IESpecs can also be used to illustrate the structure of the information exported using the IPFIX Structured Data extension [I-D.ietf-ipfix-structured-data]. Here, the semantics of the structured data elements are specified using contexts, and the information elements within each structured data element follow the structured data element, prefixed with + to show they are contained therein. Arbitrary nesting of structured data elements is possible

by using multiple + signs in the prefix. For example, a basic list of IP addresses with "one or more" semantics would be expressed using partially qualified IESpecs as follows:

```
basicList{oneOrMoreOf}  
+sourceIPv4Address(8)[4]
```

And an example subTemplateList itself containing a basicList is shown below:

```
subTemplateList{allOf}  
+basicList{oneOrMoreOf}  
++sourceIPv4Address(8)[4]  
+destinationIPv4Address(12)[4]
```

This describes a subTemplateMultilist containing all of the expressed set of source-destination pairs, where the source address itself could be one of any number in a basicList (e.g., in the case of SCTP multihoming).

The contexts associable with structured data Information Elements are the semantics, as defined in section 4.4 of [I-D.ietf-ipfix-structured-data]; a structured data Information Element without any context is taken to have undefined semantics. More information on the application of structured data is available in [I-D.ietf-ipfix-structured-data].

10. Security Considerations

The security aspects of new Information Elements must be considered in order not to give a potential attacker too much information. For example, the "A Framework for Packet Selection and Reporting" [RFC5474] concluded in section 12.3.2 that the hash functions private parameters should not be exported within IPFIX.

If some security considerations are specific to an Information Element, they MUST be mentioned in the Information Element description. For example, the ipHeaderPacketSection in the IPFIX registry mentions: "This Information Element, which may have a variable length, carries a series of octets from the start of the IP header of a sampled packet. With sufficient length, this element also reports octets from the IP payload, subject to [RFC2804]. See the Security Considerations section."

These security considerations MAY also be stressed in an accompanying Internet-Draft, as in Section 8. For example, the "Packet Sampling (PSAMP) Protocols Specification" [RFC5476] specifies: "In the basic

Packet Report, a PSAMP Device exports some number of contiguous bytes from the start of the packet, including the packet header (which includes link layer, network layer and other encapsulation headers) and some subsequent bytes of the packet payload. The PSAMP Device SHOULD NOT export the full payload of conversations, as this would mean wiretapping [RFC2804]. The PSAMP Device MUST respect local privacy laws."

11. IANA Considerations

With respect to the management of the IPFIX Information Element registry and associated subregistries located at [iana-ipfix-assignments], this document defines a process for IANA in Section 5.1, and includes a set of guidelines for IANA for applying this process in Section 4, Section 5, and Section 6.

In addition, in order to support more effective management of the Information Element lifecycle as defined in Section 5, it specifies the addition of three new columns for this registry:

Revision: a serial revision number for each Information Element, beginning at 0 for all presently existing and newly created Information Elements.

Date: the date at which the Information Element was created or last modified.

Enterprise-specific reference: for Information Elements which were deployed as enterprise-specific Information Elements for experimentation and testing, and subsequently registered in the IANA registry, specifies the private enterprise number (PEN) and IE number of the equivalent experimental IE.

12. Acknowledgements

The authors would like to acknowledge the FP7 PRISM and DEMONS projects for their material support of this work.

13. References

13.1. Normative References

[RFC3954] Claise, B., "Cisco Systems NetFlow Services Export Version 9", RFC 3954, October 2004.

- [RFC5101] Claise, B., "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information", RFC 5101, January 2008.
- [RFC5102] Quittek, J., Bryant, S., Claise, B., Aitken, P., and J. Meyer, "Information Model for IP Flow Information Export", RFC 5102, January 2008.
- [RFC5103] Trammell, B. and E. Boschi, "Bidirectional Flow Export Using IP Flow Information Export (IPFIX)", RFC 5103, January 2008.
- [RFC5610] Boschi, E., Trammell, B., Mark, L., and T. Zseby, "Exporting Type Information for IP Flow Information Export (IPFIX) Information Elements", RFC 5610, July 2009.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

13.2. Informative References

- [RFC2804] IAB and IESG, "IETF Policy on Wiretapping", RFC 2804, May 2000.
- [RFC3917] Quittek, J., Zseby, T., Claise, B., and S. Zander, "Requirements for IP Flow Information Export (IPFIX)", RFC 3917, October 2004.
- [RFC4181] Heard, C., "Guidelines for Authors and Reviewers of MIB Documents", BCP 111, RFC 4181, September 2005.
- [RFC5153] Boschi, E., Mark, L., Quittek, J., Stiemerling, M., and P. Aitken, "IP Flow Information Export (IPFIX) Implementation Guidelines", RFC 5153, April 2008.
- [RFC5470] Sadasivan, G., Brownlee, N., Claise, B., and J. Quittek, "Architecture for IP Flow Information Export", RFC 5470, March 2009.
- [RFC5471] Schmoll, C., Aitken, P., and B. Claise, "Guidelines for IP Flow Information Export (IPFIX) Testing", RFC 5471, March 2009.
- [RFC5472] Zseby, T., Boschi, E., Brownlee, N., and B. Claise, "IP

Flow Information Export (IPFIX) Applicability", RFC 5472, March 2009.

- [RFC5473] Boschi, E., Mark, L., and B. Claise, "Reducing Redundancy in IP Flow Information Export (IPFIX) and Packet Sampling (PSAMP) Reports", RFC 5473, March 2009.
- [RFC5474] Duffield, N., Chiou, D., Claise, B., Greenberg, A., Grossglauser, M., and J. Rexford, "A Framework for Packet Selection and Reporting", RFC 5474, March 2009.
- [RFC5476] Claise, B., Johnson, A., and J. Quittek, "Packet Sampling (PSAMP) Protocol Specifications", RFC 5476, March 2009.
- [RFC5477] Dietz, T., Claise, B., Aitken, P., Dressler, F., and G. Carle, "Information Model for Packet Sampling Exports", RFC 5477, March 2009.
- [RFC5655] Trammell, B., Boschi, E., Mark, L., Zseby, T., and A. Wagner, "Specification of the IP Flow Information Export (IPFIX) File Format", RFC 5655, October 2009.
- [I-D.ietf-ipfix-structured-data]
Claise, B., Dhandapani, G., Yates, S., and P. Aitken,
"Export of Structured Data in IPFIX",
draft-ietf-ipfix-structured-data-06 (work in progress),
May 2011.
- [I-D.ietf-ipfix-anon]
Boschi, E. and B. Trammell, "IP Flow Anonymization
Support", draft-ietf-ipfix-anon-06 (work in progress),
January 2011.
- [iana-ipfix-assignments]
Internet Assigned Numbers Authority, "IP Flow Information
Export Information Elements
(<http://www.iana.org/assignments/ipfix/ipfix.xml>)".

Authors' Addresses

Brian Trammell
Swiss Federal Institute of Technology Zurich
Gloriastrasse 35
8092 Zurich
Switzerland

Phone: +41 44 632 70 13
Email: trammell@tik.ee.ethz.ch

Benoit Claise
Cisco Systems, Inc.
De Kleetlaan 6a b1
1831 Diagem
Belgium

Phone: +32 2 704 5622
Email: bclaise@cisco.com

