Specification of the IPFIX Protocol for the Exchange
of IP Traffic Flow Information

Status of this Memo

Copyright Notice

Abstract

  This document specifies the IPFIX protocol that serves for
  transmitting IP traffic flow information over the network. In order

to transmit IP traffic flow information from an exporting process to
an information collecting process, a common representation of flow
data and a standard means of communicating them is required. This
document describes how the IPFIX data and templates records are
carried over a number of transport protocols from an IPFIX exporting
process to an IPFIX collecting process.

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

      Changes compared to version 24:
      EDITOR'S NOTE: this section must be removed by the RFC-editor

      -
        Remove the SCTP stream 0 restriction (transmission
       of Templates over SCTP stream zero with reliable delivery
       and the transmission of Data Records over separate streams)
      -
        Padding.
       http://www3.ietf.org/proceedings/07jul/slides/ipfix-13.pdf
      -
        Previous version of the draft didn't specifiy very clearly
       what to do for the TCP transport if a Template ID is reused within
       the template expiry time. The section 10.4.3 has been added
      -
        Included the consensus from
       http://www1.ietf.org/mail-archive/web/ipfix/current/msg03928.html
      -
        Small editorial changes

1.   Introduction

      A data network with IP traffic, primarily consists of IP Flows
      passing through the network elements of the network. It is often
      interesting, useful or even a requirement to have access to

information about these flows that pass through the network elements
   for administrative or other purposes. The IPFIX collecting process
   should be able to receive the flow information passing through
   multiple network elements within the data network. This requires

   uniformity in the method of representing the flow information and
   the means of communicating the flows from the network elements to
   the collection point. This document specifies the protocol to
   achieve these aforementioned requirements. This document specifies
   in detail the representation of different flows, the additional data
   required for flow interpretation, packet format, transport
   mechanisms used, security concerns, etc.

1.1   IPFIX Documents Overview

   The IPFIX protocol 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 [IPFIX-ARCH], per the requirements defined in
   [RFC3917]. This document specifies how IPFIX data records and
   templates are carried via a number of transport protocols from IPFIX
   exporting processes to IPFIX collecting process. IPFIX has a formal
   description of IPFIX information elements, their name, type and
   additional semantic information, as specified in [IPFIX-INFO].
   Finally [IPFIX-AS] 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.

2.   Terminology

   The definitions of the basic terms like IP Traffic Flow, Exporting
   Process, Collecting Process, Observation Points, etc. are
   semantically identical with those found in the IPFIX requirements
   document [RFC3917]. Some of the terms have been expanded for more
   clarity when defining the protocol. Additional terms required for
   the protocol has also been defined. Definitions in this document
   and in [IPFIX-ARCH] are equivalent, except that definitions which
   are only relevant to the IPFIX protocol only appear here.

   The terminology summary table in Section 2.1 gives a quick overview
   of the relationships between some of the different terms defined.

Observation Point

   An Observation Point is a location in the network where IP packets
   can be observed. Examples include: a line to which a probe is
   attached, a shared medium, such as an Ethernet-based LAN, a single
   port of a router, or a set of interfaces (physical or logical) of a
   router.

Note that every Observation Point is associated with an Observation
Domain (defined below), and that one Observation Point may be a
superset of several other Observation Points. For example one
Observation Point can be an entire line card. That would be the
superset of the individual Observation Points at the line card's
interfaces.

Observation Domain

  An Observation Domain is the largest set of Observation Points for
  which Flow information can be aggregated by a Metering Process. For
  example, a router line card may be an Observation Domain if it is
  composed of several interfaces, each of which is an Observation
  Point. In the IPFIX Message it generates, the Observation Domain
  includes its Observation Domain ID, which is unique per Exporting
  Process. That way, the Collecting Process can identify the specific
  Observation Domain from the Exporter that sends the IPFIX Messages.
  Every Observation Point is associated with an Observation Domain. .
  It is RECOMMENDED that Observation Domain IDs are also unique per
  IPFIX Device.

IP Traffic Flow or Flow

  There are several definitions of the term 'flow' being used by the
  Internet community. Within the context of IPFIX we use the
  following definition:

  A Flow is defined as a set of IP packets passing an Observation
  Point in the network during a certain time interval. All packets
  belonging to a particular Flow have a set of common properties.
  Each property is defined as the result of applying a function to the
  values of:

     1. one or more packet header fields (e.g. destination IP
     address), transport header fields (e.g. destination port number),
     or application header fields (e.g. RTP header fields [RFC3550])

     2. one or more characteristics of the packet itself (e.g. number
     of MPLS labels, etc...)

     3. one or more of fields derived from packet treatment (e.g. next
     hop IP address, the output interface, etc...)

  A packet is defined to belong to a Flow if it completely satisfies
  all the defined properties of the Flow.

  This definition covers the range from a Flow containing all packets

   observed at a network interface to a Flow consisting of just a
   single packet between two applications. It includes packets
   selected by a sampling mechanism.

Flow Key

   Each of the fields which

   1. Belong to the packet header (e.g. destination IP address)

   2. Are a property of the packet itself (e.g. packet length)

   3. Are derived from packet treatment (e.g. AS number)

   and which are used to define a Flow are termed Flow Keys.

Flow Record

   A Flow Record contains information about a specific Flow that was
   observed at an Observation Point. A Flow Record contains measured
   properties of the Flow (e.g. the total number of bytes for all the
   Flow's packets) and usually characteristic properties of the Flow
   (e.g. source IP address).

Metering Process

   The Metering Process generates Flow Records. Inputs to the process
   are packet headers and characteristics observed at an Observation
   Point, and packet treatment at the Observation Point (for example
   the selected output interface).

   The Metering Process consists of a set of functions that includes
   packet header capturing, timestamping, sampling, classifying, and
   maintaining Flow Records.

   The maintenance of Flow Records may include creating new records,
   updating existing ones, computing Flow statistics, deriving further
   Flow properties, detecting Flow expiration, passing Flow Records to
   the Exporting Process, and deleting Flow Records.

Exporting Process

The Exporting Process sends Flow Records to one or more Collecting
Processes. The Flow Records are generated by one or more Metering
Processes.

Exporter

  A device which hosts one or more Exporting Processes is termed an
  Exporter.

IPFIX Device

  An IPFIX Device hosts at least one Exporting Process. It may host
  further Exporting processes and arbitrary numbers of Observation
  Points and Metering Process.

Collecting Process

  A Collecting Process receives Flow Records from one or more
  Exporting Processes. The Collecting Process might process or store
  received Flow Records, but such actions are out of scope for this
  document.

Collector

  A device which hosts one or more Collecting Processes is termed a
  Collector.

Template

  Template is a ordered sequence of <type, length> pairs, used to
  completely specify the structure and semantics of a particular set
  of information that needs to be communicated from an IPFIX Device to
  a Collector. Each Template is uniquely identifiable by means of a
  Template ID.

IPFIX Message

  An IPFIX Message is a message originating at the Exporting Process
  that carries the IPFIX records of this Exporting Process and whose
  destination is a Collecting Process. An IPFIX Message is
  encapsulated at the transport layer.

Message Header

The Message Header is the first part of an IPFIX Message, which
provides basic information about the message such as the IPFIX
version, length of the message, message sequence number, etc.

Template Record

  A Template Record defines the structure and interpretation of fields
  in a Data Record.

Data Record

  A Data Record is a record that contains values of the parameters
  corresponding to a Template Record.

Options Template Record

  An Options Template Record is a Template Record that defines the
  structure and interpretation of fields in a Data Record, including
  defining how to scope the applicability of the Data Record.

Set

  Set is a generic term for a collection of records that have a
  similar structure. In an IPFIX Message, one or more Sets follow the
  Message Header.

  There are three different types of Sets: Template Set, Options
  Template Set, and Data Set.

Template Set

  A Template Set is a collection of one or more Template Records that
  have been grouped together in an IPFIX Message.

Options Template Set

  An Options Template Set is a collection of one or more Options
  Template Records that have been grouped together in an IPFIX
  Message.

Data Set

  A Data Set is one or more Data Records, of the same type, that are
  grouped together in an IPFIX Message. Each Data Record is

   previously defined by a Template Record or an Options Template
   Record.

Information Element

   An Information Element is a protocol and encoding independent
   description of an attribute which may appear in an IPFIX Record.
   The IPFIX information model [IPFIX-INFO] defines the base set of
   Information Elements for IPFIX. The type associated with an
   Information Element indicates constraints on what it may contain and
   also determines the valid encoding mechanisms for use in IPFIX.

Transport Session

   In SCTP, the transport session is known as the SCTP association,
   which is uniquely identified by the SCTP endpoints [RFC2960]; in TCP,
   the transport session is known as the TCP connection, which is
   uniquely identified by the combination of IP addresses and TCP ports
   used; In UDP, the transport session is known as the UDP session,
   which is uniquely identified by the combination of IP addresses and
   UDP ports used.

2.1   Terminology Summary Table

```
   +-----------------+---------------------------------------------+
   |                 |                  contents                   |
   |                 +------------------+--------------------------+
   |      Set        |     Template     |         record           |
   +-----------------+------------------+--------------------------+
   |     Data Set    |         /        |      Data Record(s)      |
   +-----------------+------------------+--------------------------+
   |   Template Set  | Template Record(s) |          /             |
   +-----------------+------------------+--------------------------+
   | Options Template | Options Template |          /             |
   |       Set       | Record(s)        |                          |
   +-----------------+------------------+--------------------------+
```

      Figure A: Terminology Summary Table

   A Data Set is composed of Data Record(s). No Template Record is
   included. A Template Record or an Options Template Record defines
   the Data Record.

   A Template Set contains only Template Record(s).

   An Options Template Set contains only Options Template Record(s).

3.   IPFIX Message Format

   An IPFIX Message consists of a Message Header followed by one or
   more Sets. The Sets can be any of the possible three types: Data
   Set, Template Set or Options Template Set.

   The format of the IPFIX Message is shown in Figure B.

```
      +------------------------------------------------------+
      | Message Header                                       |
      +------------------------------------------------------+
      | Set                                                  |
      +------------------------------------------------------+
      | Set                                                  |
      +------------------------------------------------------+
        ...
      +------------------------------------------------------+
      | Set                                                  |
      +------------------------------------------------------+
```
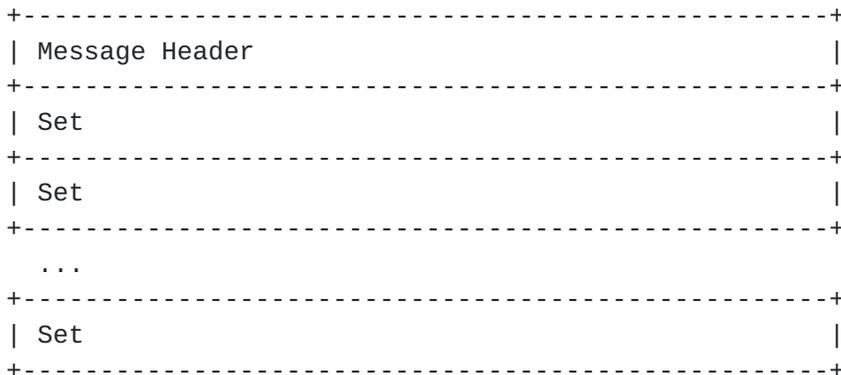
          Figure B: IPFIX Message format

   The Exporter MUST code all binary integers of the Message Header and
   the different Sets in network byte order (also known as the big-
   endian byte ordering).

   Following are some examples of IPFIX Messages:

   1. An IPFIX Message consisting of interleaved Template, Data, and
   Options Template Sets - A newly created Template is exported as soon
   as possible. So if there is already an IPFIX Message with a Data
   Set that is being prepared for export, the Template and Option
   Template Sets are interleaved with this information, subject to
   availability of space.

```
   +--------+----------------------------------------------------------+
   |        | +----------+ +---------+     +-----------+ +---------+ |
   |Message | | Template | | Data    |     | Options   | | Data    | |
   | Header | | Set      | | Set     | ... | Template  | | Set     | |
   |        | |          | |         |     | Set       | |         | |
   |        | +----------+ +---------+     +-----------+ +---------+ |
```

```
   +--------+----------------------------------------------------------+
```

        Figure C: IPFIX Message example 1

2. An IPFIX Message consisting entirely of Data Sets - After the
appropriate Template Records have been defined and transmitted to
the Collecting Process, the majority of IPFIX Messages consist
solely of Data Sets.

```
   +--------+------------------------------------------------+
   |        | +---------+     +---------+       +---------+ |
   |Message | | Data    |     | Data    |       | Data    | |
   | Header | | Set     | ... | Set     | ...   | Set     | |
   |        | +---------+     +---------+       +---------+ |
   +--------+------------------------------------------------+
```

        Figure D: IPFIX Message example 2

3. An IPFIX Message consisting entirely of Template and Options
Template Sets.

```
   +--------+--------------------------------------------------+
   |        | +----------+     +----------+       +----------+ |
   |Message | | Template |     | Template |       | Options  | |
   | Header | | Set      | ... | Set      | ...   | Template | |
   |        | |          |     |          |       | Set      | |
   |        | +----------+     +----------+       +----------+ |
   +--------+--------------------------------------------------+
```

        Figure E: IPFIX Message example 3

3.1   Message Header Format

   The format of the IPFIX Message Header is shown in Figure F.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |       Version Number          |            Length             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                           Export Time                         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                         Sequence Number                       |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
|                      Observation Domain ID                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

   Figure F: 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

        Time in seconds since 0000 UTC Jan 1st 1970, at which the
        IPFIX Message Header leaves the Exporter.

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.

3.2   Field Specifier Format

   Vendors need the ability to define proprietary Information Elements,
   because, for example, they are delivering a pre-standards product,
   or the Information Element is in some way commercially sensitive.
   This section describes the Field Specifier format for both IETF
   specified Information Elements [IPFIX-INFO] and enterprise-specific
   Information Elements.

   The Information Elements are identified by the Information Element
   identifier. When the Enterprise bit is set to 0, the corresponding
   Information Element identifier will report an IETF specified
   Information Element, and the Enterprise Number MUST NOT be present.
   When the Enterprise bit is set to 1, the corresponding Information
   Element identifier will report an enterprise-specific Information
   Element and the Enterprise Number MUST be present. An example of
   this is shown in section 13.4.2

   The Field Specifier format is shown in Figure G.

```
      0                   1                   2                   3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |E|  Information Element ident. |        Field Length           |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                      Enterprise Number                        |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
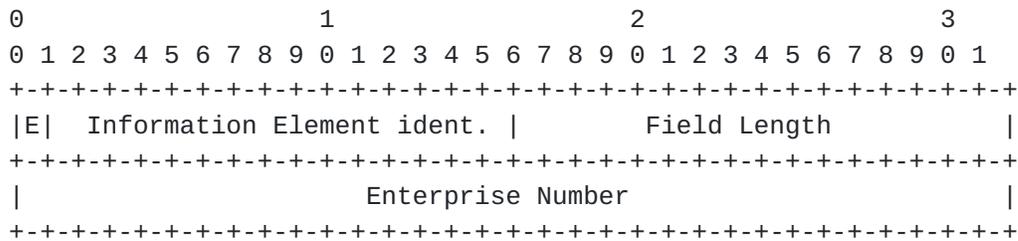
        Figure G: Field Specifier format

    Where:

   E
        Enterprise bit. This is the first bit of the Field
        Specifier. If this bit is zero, the Information Element
        Identifier identifies an IETF specified Information Element,
        and the four octet Enterprise Number field MUST NOT be
        present. If this bit is one, the Information Element
        identifier identifies an enterprise-specific Information
        Element, and the Enterprise Number filed MUST be present.

   Information Element identifier
       A numeric value that represents the type of the Information
       Element. Refer to [IPFIX-INFO].

   Field Length
       The length of the corresponding encoded Information Element,
       in octets. Refer to [IPFIX-INFO]. The field length may be
       smaller than the definition in [IPFIX-INFO] if reduced size
       encoding is used (see section 6.2). The value 65535 is
       reserved for variable length Information Element (see
       section 7).

   Enterprise Number
       IANA enterprise number [PEN] of the authority defining the
       Information Element identifier in this Template Record.

3.3   Set and Set Header Format

  A Set is a generic term for a collection of records that have a
  similar structure. There are three different types of Sets:
  Template Sets, Options Template Sets, and Data Sets. Each of these
  Sets consists of a Set Header and one or more Records. The Set
  Format and the Set Header Format are defined in the following
  sections.

3.3.1   Set Format

  A Set has the format shown in figure H. The records types can be
  either Template Records, Options Template Records or Data Records.
  The record types MUST NOT be mixed within a Set.

```
        +--------------------------------------------------+
        | Set Header                                       |
        +--------------------------------------------------+
        | record                                           |
        +--------------------------------------------------+
        | record                                           |
        +--------------------------------------------------+
         ...
        +--------------------------------------------------+
        | record                                           |
        +--------------------------------------------------+
        | Padding (opt.)                                   |
        +--------------------------------------------------+
```

                 Figure H: Set Format

     The Set Field Definitions are as follows:

     Set Header
          The Set Header Format is defined in section 3.3.2.

     Record
          One of the Record Formats: Template Record or Options Template
          Record or Data Record Format.

     Padding
          The Exporting Process MAY insert some padding octets, so that
          the subsequent Set starts at an aligned boundary. For
          security reasons, the padding octet(s) MUST be composed of
          zero (0) valued octets. The padding length MUST be shorter
          than any allowable Record in this Set. If padding of the
          IPFIX Message is desired in combination with very short
          Records, then the padding Information Element 'paddingOctets'
          [IPFIX-INFO] can be used for padding Records such that their
          length is increased to a multiple of 4 or 8 octets. Because
          Template Sets are always 4-octet aligned by definition,
          padding is only needed in case of other alignments e.g. on 8-
          octet boundaries.

3.3.2   Set Header Format

  Every Set contains a common header. This header is defined in figure
  I.

      0   1   2   3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
     +-+
     | Set ID | Length |
     +-+
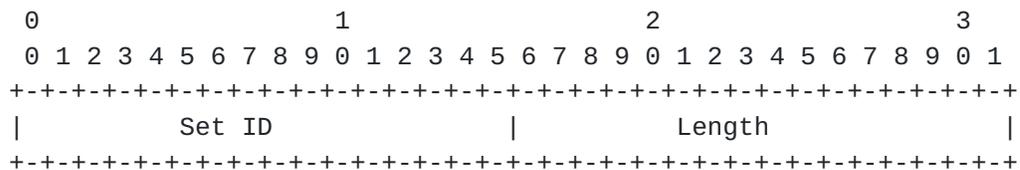
                  Figure I: Set Header Format

  The Set Header Field Definitions are as follows:

     Set ID
        Set ID value identifies the Set. A value of 2 is reserved for
        the Template Set. A value of 3 is reserved for the Option
        Template Set. All other values from 4 to 255 are reserved

      for future use. Values above 255 are used for Data Sets. The
      Set ID values of 0 and 1 are not used for historical reasons
      [RFC3954].

   Length
        Total length of the Set in octets including the Set Header,
        all records and the optional padding. Because an individual
        Set MAY contain multiple records, the Length value MUST be
        used to determine the position of the next Set.

3.4   Record Format

   IPFIX defines three record formats, defined in the next sections:
   the Template Record Format, the Options Template Record Format and
   the Data Record Format.

3.4.1   Template Record Format

   One of the essential elements in the IPFIX record format is the
   Template Record. Templates greatly enhance the flexibility of the
   record format because they allow the Collecting Process to process
   IPFIX Messages without necessarily knowing the interpretation of all
   Data Records. A Template Record contains any combination of IANA-
   assigned and/or enterprise-specific Information Elements
   identifiers.

   The format of the Template Record is shown in Figure J. It consists
   of a Template Record Header and one or more Field Specifiers. The
   definition of the Field Specifiers is given in figure G above.

```
        +--------------------------------------------------+
        | Template Record Header                           |
        +--------------------------------------------------+
        | Field Specifier                                  |
        +--------------------------------------------------+
        | Field Specifier                                  |
        +--------------------------------------------------+
         ...
        +--------------------------------------------------+
        | Field Specifier                                  |
        +--------------------------------------------------+
```
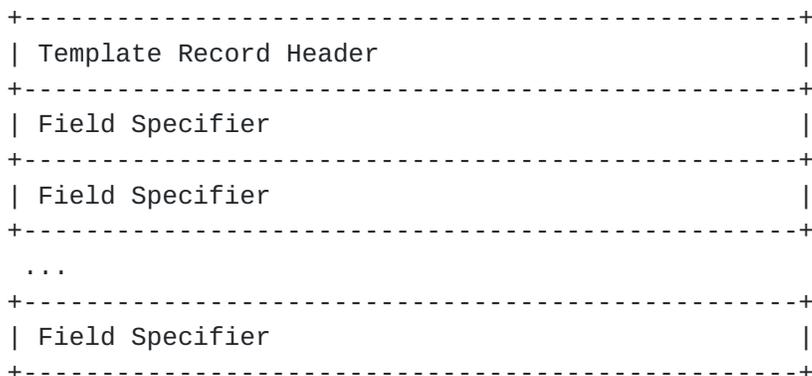
              Figure J: Template Record Format

The format of the Template Record Header is shown in Figure K.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |       Template ID (> 255)     |         Field Count           |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
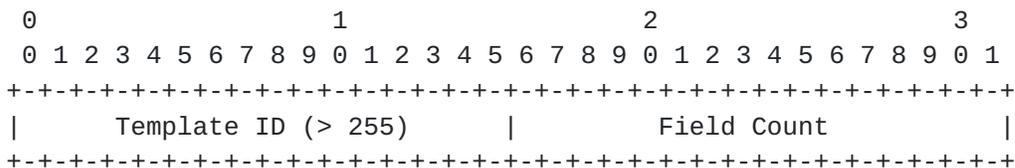
              Figure K: Template Record Header Format

The Template Record Header Field Definitions are as follows:

   Template ID
      Each of the newly generated Template Records is given a unique
      Template ID. This uniqueness is local to the Transport
      Session and Observation Domain that generated the Template ID.
      Template IDs 0-255 are reserved for Template Sets, Options
      Template Sets, and other reserved Sets yet to be created.
      Template IDs of Data Sets are numbered from 256 to 65535.
      There are no constraints regarding the order of the Template
      ID allocation.

   Field Count
      Number of fields in this Template Record.

The example in Figure L shows a Template Set with mixed standard and
enterprise-specific Information Elements. It consists of Set Header,
Template Header and several Field Specifiers.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |           Set ID = 2          |             Length            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |        Template ID = 256      |         Field Count = N       |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |1| Information Element id. 1.1 |        Field Length 1.1       |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                      Enterprise Number  1.1                   |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |0| Information Element id. 1.2 |        Field Length 1.2       |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |              ...              |              ...              |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |1| Information Element id. 1.N |        Field Length 1.N       |
```

```
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                      Enterprise Number  1.N                  |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |      Template ID = 257      |          Field Count = M       |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |0| Information Element id. 2.1 |       Field Length 2.1        |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |1| Information Element id. 2.2 |       Field Length 2.2        |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                      Enterprise Number  2.2                  |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |            ...              |              ...               |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |1| Information Element id. 2.M |       Field Length 2.M        |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                      Enterprise Number  2.M                  |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                         Padding (opt)                        |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

              Figure L: Template Set Example

   Information Element Identifiers 1.2 and 2.1 are defined by the IETF
   (Enterprise bit = 0) and therefore do not need an Enterprise Number
   to identify them.

3.4.2   Options Template Record Format

   Thanks to the notion of scope, The Options Template Record gives the
   Exporter the ability to provide additional information to the
   Collector which would not be possible with Flow Records alone.

   One Options Template Record example is the "Flow Keys", which
   reports the Flow Keys for a Template, which is defined as the scope.
   Another example is the "Template configuration", which reports the
   configuration sampling parameter(s) for the Template, which is
   defined as the scope.

3.4.2.1 Scope

   The scope, which is only available in the Options Template Set,
   gives the context of the reported Information Elements in the Data
   Records. Note that the IPFIX Message Header already contains the

   Observation Domain ID (the identifier of the Observation Domain). If
   not zero, this Observation Domain ID can be considered as an
   implicit scope for the Data Records in the IPFIX Message. The
   Observation Domain ID MUST be zero when the IPFIX Message contains
   data records with different Observation Domain ID values defined as
   scopes.

   Multiple scope fields MAY be present in the Options Template Record,
   in which case, the composite scope is the combination of the scopes.
   For example, if the two scopes are defined as "metering process" and
   "template", the combined scope is this Template for this metering
   process. The order of the scope fields, as defined in the Options
   Template Record, is irrelevant in this case. However, if the order
   of the scope fields in the Options Template Record is relevant, the
   order of the scope fields MUST be used. For example, if the first
   scope defines the filtering function, while the second scope defines
   the sampling function, the order of the scope is important. Applying
   the sampling function first, followed by the filtering function,
   would lead to potentially different Data Records than applying the
   filtering function first, followed by the sampling function. In
   this case, the Collector deduces the function order by looking at
   the order of the scope in the Options Template Record.

   The scope is an Information Element specified in the IPFIX
   Information Model [IPFIX-INFO]. An IPFIX compliant implementation
   of the Collecting Process SHOULD support this minimum set of
   Information Elements as scope: LineCardId, TemplateId,
   exporterIPv4Address, exporterIPv6Address, and ingressInterface.
   Note that other Information Elements such as meteringProcessId,
   exportingProcessId, observationDomainId, etc. are also valid scopes.
   The IPFIX protocol doesn't prevent the use of any Information
   Elements for scope. However some Information Element types don't
   make sense if specifed as scope. For example: the counter
   Information Elements.

   Finally, note that the Scope Field Count MUST NOT be zero.

3.4.2.2 Options Template Record Format

   An Options Template Record contains any combination of IANA-assigned
   and/or enterprise-specific Information Elements identifiers.

   The format of the Options Template Record is shown in Figure M. It
   consists of an Options Template Record Header and one or more Field

Specifiers. The definition of the Field Specifiers is given in
figure G above.

```
+----------------------------------------------------+
| Options Template Record Header                     |
+----------------------------------------------------+
| Field Specifier                                    |
+----------------------------------------------------+
| Field Specifier                                    |
+----------------------------------------------------+
  ...
+----------------------------------------------------+
| Field Specifier                                    |
+----------------------------------------------------+
```
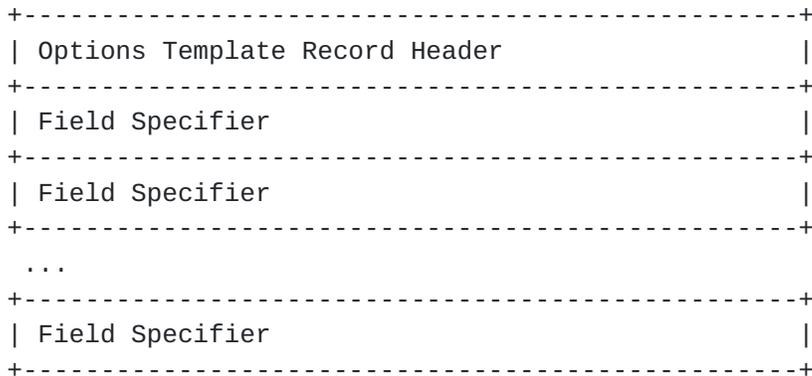
           Figure M: Options Template Record Format

The format of the Options Template Record Header is shown in Figure
N.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Template ID (> 255)   |          Field Count          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Scope Field Count        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
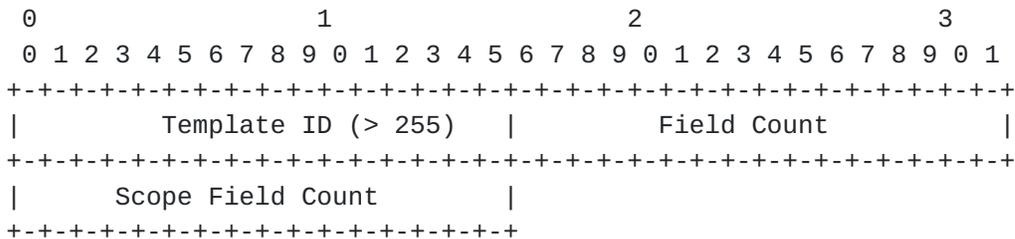
         Figure N: Options Template Record Header Format

The Options Template Record Header Field Definitions are as follows:

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

    Field Count
       Number of all fields in this Options Template Record,
       including the Scope Fields.

    Scope Field Count
       Number of scope fields in this Options Template Record. The
       Scope Fields are normal Fields except that they are

          interpreted as Scope at the Collector. The Scope Field Count
          MUST NOT be zero.

     The example in Figure O shows an Option Template Set with mixed IETF
     and enterprise-specific Information Elements. It consists of Set
     Header, Option Template Header and several Field Specifiers.

```
    0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |          Set ID = 3           |             Length            |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |         Template ID = 258     |        Field Count = N + M    |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |      Scope Field Count = N    |0|  Scope 1 Infor. Element Id. |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |      Scope 1 Field Length     |0|  Scope 2 Infor. Element Id. |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |      Scope 2 Field Length     |             ...               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |            ...                |1|  Scope N Infor. Element Id. |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |      Scope N Field Length     |   Scope N Enterprise Number ...
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    ...  Scope N Enterprise Number  |1| Option 1 Infor. Element Id. |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |     Option 1 Field Length     |  Option 1 Enterprise Number ...
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    ... Option 1 Enterprise Number  |             ...               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |             ...               |0| Option M Infor. Element Id. |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |     Option M Field Length     |      Padding (optional)       |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

                  Figure O: Option Template Set Example

3.4.3   Data Record Format

     The Data Records are sent in Data Sets. The format of the Data
     Record is shown in Figure P. It consists only of one or more Field
     Values. The Template ID to which the Field Values belong is encoded
     in the Set Header field "Set ID" i.e., "Set ID" = "Template ID".

```
       +---------------------------------------------------+
       | Field Value                                       |
       +---------------------------------------------------+
       | Field Value                                       |
       +---------------------------------------------------+
        ...
       +---------------------------------------------------+
       | Field Value                                       |
       +---------------------------------------------------+
```
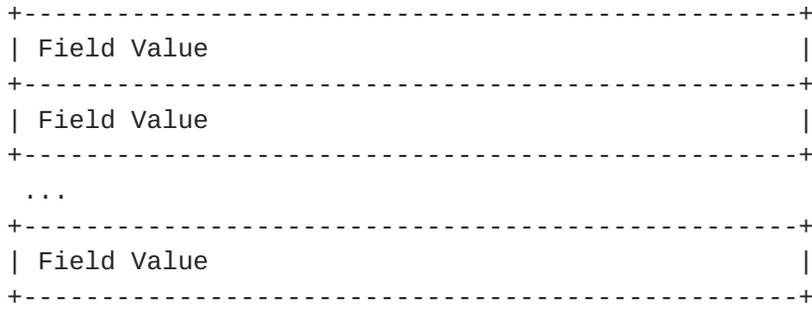
              Figure P: Data Record Format

Note that Field Values do not necessarily have a length of 16 bits.
Field Values are encoded according to their data type specified in
[IPFIX-INFO].

Interpretation of the Data Record format can be done only if the
Template Record corresponding to the Template ID is available at the
Collecting Process.

The example in Figure Q shows a Data Set. It consists of a Set
Header several Field Values.

```
       0                   1                   2                   3
       0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |    Set ID = Template ID       |            Length             |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |    Record 1 - Field Value 1   |   Record 1 - Field Value 2    |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |     Record 1 - Field Value 3  |             ...               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |    Record 2 - Field Value 1   |   Record 2 - Field Value 2    |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |     Record 2 - Field Value 3  |             ...               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |     Record 3 - Field Value 1  |   Record 3 - Field Value 2    |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |     Record 3 - Field Value 3  |             ...               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                 ...           |      Padding (optional)       |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
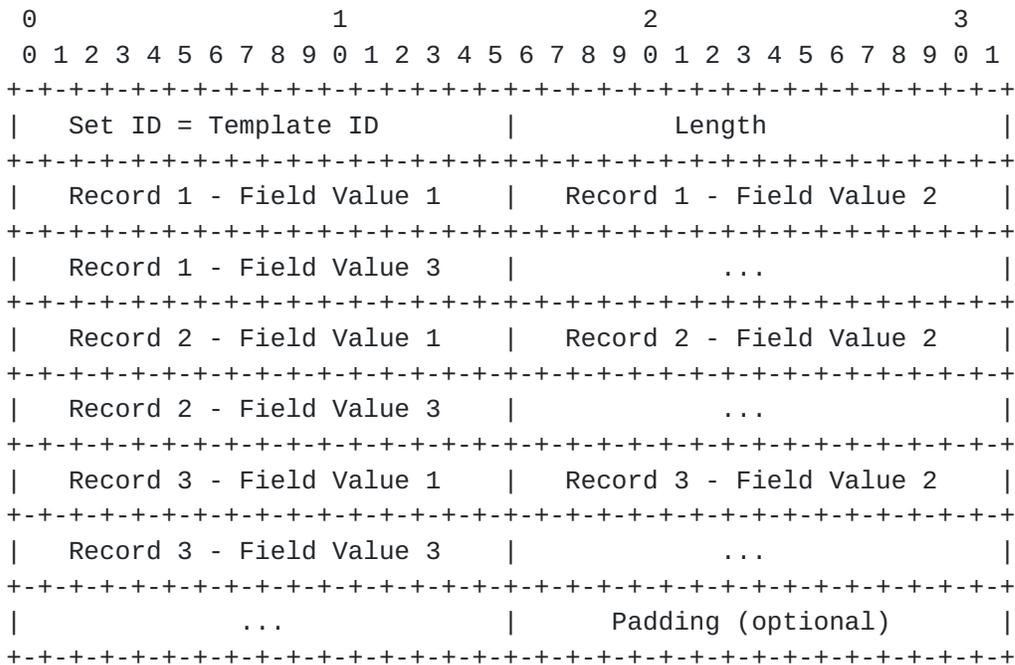
         Figure Q: Data Set, containing Data Records

4.   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 Option Template and Options Template Records defined in these
   sub-sections, which impose some constraints on the Metering Process
   and Exporting Process implementations, MAY be implemented. If
   implemented, the specific Option Templates SHOULD be implemented as
   specified in these sub-sections.

   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.

4.1   The Metering Process Statistics Option Template

   The Metering Process Statistics Option Template specifies the
   structure of a Data Record for reporting Metering Process
   statistics. It SHOULD contain the following Information Elements
   that are defined in [IPFIX-INFO]:

        observationDomainId   An identifier of an Observation Domain
                                 that is locally unique to the Exporting
                                 Process. This Information Element MUST
                                 be defined as a Scope Field.

        exportedMessageTotalCount
                                 The total number of IPFIX Messages
                                 that the Exporting Process successfully
                                 sent to the Collecting Process since
                                 the Exporting Process re-
                                 initialization.

        exportedFlowTotalCount   The total number of Flow Records that
                                 the Exporting Process successfully sent
                                 to the Collecting Process since the
                                 Exporting Process re-initialization.

        exportedOctetTotalCount The total number of octets that the
                                 Exporting Process successfully sent to

                              the Collecting Process since the
                              Exporting Process re-initialization.

   The Exporting Process SHOULD export the Data Record specified by the
   Metering Process Statistics Option Template on a regular basis or
   based on some export policy. This periodicity or export policy
   SHOULD be configurable.

   Note that if several Metering Processes are available on the
   Exporter Observation Domain, the Information Element
   meteringProcessId MUST be specified as an additional Scope Field.

4.2   The Metering Process Reliability Statistics Option Template

   The Metering Process Reliability Option Template specifies the
   structure of a Data Record for reporting lack of reliability in the
   Metering Process. It SHOULD contain the following Information
   Elements that are defined in [IPFIX-INFO]:

        observationDomainId   An identifier of an Observation Domain
                               that is locally unique to the Exporting
                               Process. This Information Element MUST
                               be defined as a Scope Field.

        ignoredPacketTotalCount The total number of IP packets that the
                               Metering Process did not process.

        ignoredOctetTotalCount The total number of octets in observed
                               IP packets that the Metering Process
                               did not process.

        time first ignored   The time stamp of the first IP packet
                               that was ignored by the Metering
                               Process. For this time stamp, any of
                               the "flowStart" time stamp Information
                               Elements flowStartMilliseconds,
                               flowStartMicroseconds,
                               flowStartNanoseconds, and
                               flowStartDeltaMicroseconds can be used.

        time last ignored   The time stamp of the last IP packet
                               that was ignored by the Metering
                               Process. For this time stamp, any of
                               the "flowEnd" time stamp Information

                               Elements flowEndMilliseconds,
                               flowEndMicroseconds,
                               flowEndNanoseconds, and
                               flowEndDeltaMicroseconds can be used.

   The Exporting Process SHOULD export the Data Record specified by the
   Metering Process Reliability Statistics Option Template on a regular
   basis or based on some export policy. This periodicity or export
   policy SHOULD be configurable.

   Note that if several Metering Processes are available on the
   Exporter Observation Domain, the Information Element
   meteringProcessId MUST be specified as an additional Scope Field.

4.3   The Exporting Process Reliability Statistics Option Template

   The Exporting Process Reliability Option Template specifies the
   structure of a Data Record for reporting lack of reliability in the
   Exporting process. It SHOULD contain the following Information
   Elements that are defined in [IPFIX-INFO]:

        Exporting Process ID   The identifier of the Exporting Process
                                  for which lack of reliability is
                                  reported. There are three Information
                                  Elements specified in [IPFIX-INFO] that
                                  can be used for this purpose:
                                  exporterIPv4Address,
                                  exporterIPv6Address, or
                                  exportingProcessId. This
                                  Information Element MUST be defined
                                  as a Scope Field.

        notSentFlowTotalCount   The total number of Flows that were
                                  generated by the Metering Process and
                                  but dropped by the Metering Process or
                                  by the Exporting Process instead of
                                  sending it to the Collecting Process.

        notSentPacketTotalCount The total number of packets in Flow
                                  Records that were generated by the
                                  Metering Process and but dropped by
                                  the Metering Process or by the
                                  Exporting Process instead of sending
                                  it to the Collecting Process.

           notSentOctetTotalCount The total number of octets in packets
                                    in Flow Records that were generated by
                                    the Metering Process and but dropped
                                    by the Metering Process or by the
                                    Exporting Process instead of sending
                                    it to the Collecting Process.

           time first flow dropped The time stamp of the first Flow
                                    was dropped by the Metering
                                    Process. For this time stamp, any of
                                    the "flowStart" time stamp Information
                                    Elements flowStartMilliseconds,
                                    flowStartMicroseconds,
                                    flowStartNanoseconds, and
                                    flowStartDeltaMicroseconds can be used.

           time last flow dropped The time stamp of the last IP packet
                                    that was ignored by the Metering
                                    Process. For this time stamp, any of
                                    the "flowEnd" time stamp Information
                                    Elements flowEndMilliseconds,
                                    flowEndMicroseconds,
                                    flowEndNanoseconds, and
                                    flowEndDeltaMicroseconds can be used.

   The Exporting Process SHOULD export the Data Record specified by the
   Exporting Process Reliability Statistics Option Template on a
   regular basis or based on some export policy. This periodicity or
   export policy SHOULD be configurable.

4.4   The Flow Keys Option Template

   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 server as
   Flow Keys of the reported Flow.

   The Flow Keys Option Template SHOULD contain the following
   Information Elements that are defined in [IPFIX-INFO]:

       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.

5.   IPFIX Message Header "Export Time" and Flow Record Time

   The IPFIX Message Header "Export Time" field is the time in seconds
   since 0000 UTC Jan 1st, 1970, at which the IPFIX Message Header
   leaves the Exporter. The time-related Information Elements
   specified in [IPFIX-INFO] MAY use this "Export Time" as base time
   and specify an offset relative to it, instead of using a common base
   time, such as 0000 UTC Jan 1st, 1970. All Information Elements that
   do not have their base time defined by their data type, MUST have
   the base time clearly specified in their description.

   For example, Data Records requiring a microsecond precision can
   export the flow start and end times with the flowStartMicroseconds
   and flowEndMicroseconds Information Elements [IPFIX-INFO],
   containing the time since 0000 UTC Jan 1st 1970. An alternate
   solution is to export the flowStartDeltaMicroseconds and
   flowEndDeltaMicroseconds Information Elements [IPFIX-INFO] in the
   Data Record, which respectively report the flow start and end time
   offsets compared to the IPFIX Message Header "Export Time". The
   latter solution lowers the export bandwidth requirement while it
   increases the load on the Exporter as the Exporting Process must
   calculate the flowStartDeltaMicroseconds and
   flowEndDeltaMicroseconds of every single Data Record before
   exporting the IPFIX Message.

   It must be noted that using time-related Information Elements with
   offset times compared to the IPFIX Message Header "Export Time"
   imposes some time constraints on the Data Records contained in the
   IPFIX Message. In the example of flowStartDeltaMicroseconds and
   flowEndDeltaMicroseconds Information Elements [IPFIX-INFO], the Data
   Record must be exported within a maximum of 71 minutes after its
   creation. Otherwise, the 32-bits counter would not be sufficient to
   contain the flow start time offset.

6.   Linkage with the Information Model

   The Information Elements [IPFIX-INFO] MUST be sent in canonical
   format in network byte order (also known as the big-endian byte
   ordering).

6.1   Encoding of IPFIX Data Types

   The following sections will define the encoding of the data types
   specified in [IPFIX-INFO].

6.1.1   Integral Data Types

   Integral data types - octet, signed8, unsigned16, signed16,
   unsigned32, signed32, signed64 and unsigned64 - MUST be encoded
   using the default canonical format in network byte order. Signed
   Integral data types are represented in two's complement notation.

6.1.2   Address Types

   Address types - macAddress, ipv4Address and ipv6Address - MUST be
   encoded the same way as the integral data types. The macAddress is
   treated as a 6-octet integer, the ipv4Address as a 4-octet integer
   and the ipv6Address as a 16-octet integer.

6.1.3   float32

   The float32 data type MUST be encoded as an IEEE single-precision 32-
   bit floating point-type, as specified in [IEEE.754.1985].

6.1.4   float64

   The float64 data type MUST be encoded as an IEEE double-precision
   64-bit floating point-type, as specified in [IEEE.754.1985].

6.1.5   boolean

   The boolean data type is specified according to the TruthValue in
   [RFC2579]: that is an integer with the value 1 for true and a value
   2 for false. Every other value is undefined. The boolean data type
   MUST be encoded in a single octet.

6.1.6   string and octetarray

   The data type string represents a finite length string of valid

   characters of the Unicode character encoding set. The string data
   type MUST be encoded in UTF-8 format. The string is sent as an
   array of octets using an information element of fixed or variable
   length. The length of the information element specifies the length
   of the octetarray.

6.1.7   dateTimeSeconds

   The data type dateTimeseconds represents a time value in units of
   seconds normalised to the GMT timezone. It MUST be encoded in a 32-
   bit integer containing the number of seconds since 0000 UTC Jan 1st
   1970. The 32-bit integer allows the time encoding up to 136 years.

6.1.8   dateTimeMilliseconds

   The data type dateTimeMilliseconds represents a time value in units
   of milliseconds normalized to the GMT timezone. It MUST be encoded
   in a 64-bit integer containing the number of milliseconds since 0000
   UTC Jan 1st 1970.

6.1.9   dateTimeMicroseconds

   The data type dateTimeMicroseconds represents a time value in units
   of microseconds normalized to the GMT timezone. It MUST be encoded
   in a 64-bit integer according to the NTP format given in [RFC1305].

6.1.10 dateTimeNanoseconds

   The data type of dateTimeNanoseconds represents a time value in
   units of nanoseconds normalized to the GMT timezone. It MUST be
   encoded in a 64-bit integer according to the NTP format given in
   [RFC1305].

6.2   Reduced Size Encoding of Integer and Float Types

   Information Elements containing integer, string, float, and
   octetarray types in the information model MAY be encoded using
   fewer octets than those implied by their type in the information
   model definition [IPFIX-INFO], based on the assumption that the
   smaller size is sufficient to carry any value the Exporter may need
   to deliver. This reduces the network bandwidth requirement
   between the Exporter and the Collector. Note that the Information

   Element definitions [IPFIX-INFO] will always define the maximum
   encoding size.

   For instance the information model [IPFIX-INFO] defines byteCount as
   an unsigned64 type, which would require 64-bits. However if the
   Exporter will never locally encounter the need to send a value
   larger than 4294967295, it may chose to send the value instead as an
   unsigned32. For example, a core router would require an unsigned64
   byteCount while an unsigned32 might be sufficient for an access
   router.

   This behavior is indicated by the Exporter by specifying a type size
   with a smaller length than that associated with the assigned type of
   the Information Element. In the example above the Exporter would
   place a length of 4 versus 8 in the Template.

   If reduced sizing is used, it MUST only be applied to the following
   integer types: unsigned64, signed64, unsigned32, signed32,
   unsigned16, signed16. The signed versus unsigned property of the
   reported value MUST be preserved. The reduction in size can be to
   any number of octets smaller than the original type if the data
   value still fits, i.e. so that only leading zeroes are dropped. For
   example, an unsigned64 can be reduced in size to 7, 6, 5, 4, 3, 2,
   or 1 octet(s).

   Reduced sizing can also be used to reduce float64 to float32. The
   float32 not only has a reduced number range, but due to the smaller
   mantissa is also less precise.

   The reduced size encoding MUST NOT be applied to
   dateTimeMicroseconds or to dateTimeNanoseconds because these
   represent an inherent structure that would be destroyed by using
   less than the original number of bytes.

7.   Variable Length Information Element

   The IPFIX Template mechanism is optimized for fixed length
   Information Elements [IPFIX-INFO]. Where an Information Element has
   a variable length the following mechanism MUST be used to carry the
   length information, for both the IETF and proprietary Information
   Elements.

   In the Template Set the Information Element Field Length is recorded
   as 65535. This reserved length value notifies the Collecting
   Process that length of the Information Element will be carried in
   the Information Element content itself.

   In most cases the length of the Information Element will be less
   than 255 octets. The following length encoding mechanism optimizes
   the overhead of carrying the Information Element length in this
   majority case. The length is carried in the octet before the
   Information Element, as shown in Figure R.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | Length (< 255)|          Information element                 |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                      ... continuing as needed                |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
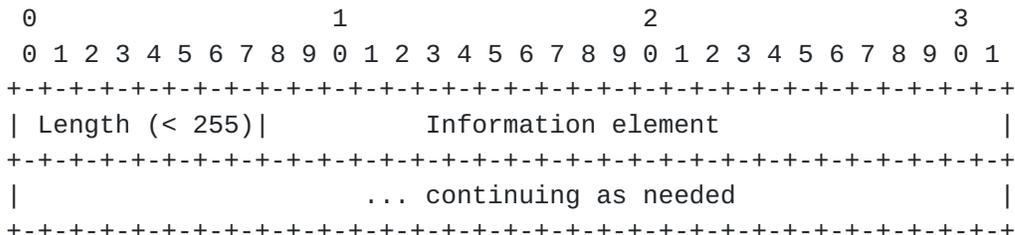
    Figure R: Variable Length Information Element (length < 255 octets)

   If the length of the Information Element is greater than or equal to
   255 octets, the length is encoded into 3 octets before the
   Information Element. The first octet is 255 and the length is
   carried in the second and third octets, as shown in Figure S.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |      255      |       Length (0 to 65535)     |      IE      |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                      ... continuing as needed                |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
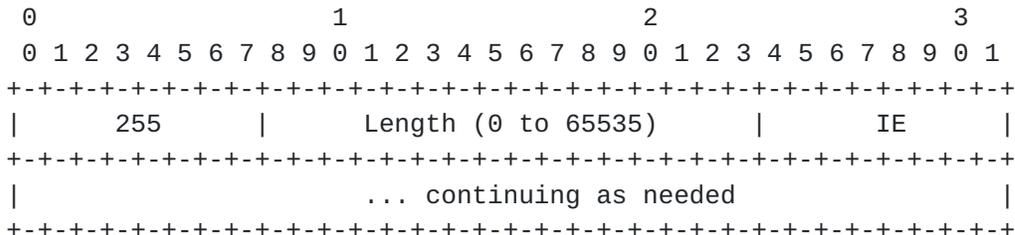
      Figure S: Variable Length Information Element
             (length 0 to 65535) octets

   The octets carrying the length (either the first or the first three
   octets) MUST NOT be included in the length of the Information
   Element.

8.   Template Management

   This section describes Template management when using SCTP and PR-
   SCTP as the transport protocol. Any necessary changes to Template
   management specifically related to TCP or UDP transport protocols are
   specified in section 10.

The Exporting Process assigns and maintains the Template IDs per SCTP
association for the Exporter's Observation Domains. A newly created
Template Record is assigned an unused Template ID by the Exporting
Process.

If a specific Information Element is required by a Template but is
not available in observed packets, the Exporting Process MAY choose
to export Flow Records without this Information Element in a Data
Record defined by a new Template.

If an Information Element is required more than once in Template, the
different occurrences of this Information Element SHOULD follow the
logical order of their treatments by the Metering Process. For
example, if a selected packet goes through two hash functions, and if
the two hash values are sent within a single Template, the first
occurrence of the hash value should belong to the first hash function
in the Metering Process. For example, when exporting the two source
IP addresses of an IPv4 in IPv4 packets, the first sourceIPv4Address
Information Element occurrence should be the IPv4 address of the
outer header, while the second occurrence should be the inner header
one.

Template Sets and Options Template Sets may be sent on any SCTP
stream. Template Sets and Options Template Sets MUST be sent
reliably, using SCTP ordered delivery. As such, the Collecting
Process MUST store the Template Record information for the duration
of the SCTP association so that it can interpret the corresponding
Data Records that are received in subsequent Data Sets.

The Exporting Process SHOULD transmit the Template Set and Options
Template Set in advance of any Data Sets that use that (Options)
Template ID, to help ensure that the Collector has the Template
Record before receiving the first Data Record. Data Records that
correspond to a Template Record MAY appear in the same and/or
subsequent IPFIX Message(s).

Different Observation Domains from the same SCTP association may use
the same Template ID value to refer to different Templates.

The Templates that are not used anymore SHOULD be deleted. Before
reusing a Template ID, the Template MUST be deleted. In order to
delete an allocated Template, the Template is withdrawn through the
use of a Template Withdraw Message.

The Template Withdraw Message MUST NOT be sent until sufficient time
has elapsed to allow the Collecting Process to receive and process
the last Data Record using this Template information. This time MUST
be configurable. A suitable default value is 5 seconds after the
last Data Record has been sent.

The Template ID from a withdrawn Template MUST NOT be reused until
sufficient time has elapsed to allow for the Collecting Process to
receive and process the Template withdraw message.

A Template Withdraw Message is a Template Record for that Template ID
with a Field Count of 0. The format of the Template Withdrawal
Message is shown in figure T.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Set ID = (2 or 3)      |         Length = 16           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Template ID N        |         Field Count = 0       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Template ID ...      |         Field Count = 0       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Template ID M        |         Field Count = 0       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
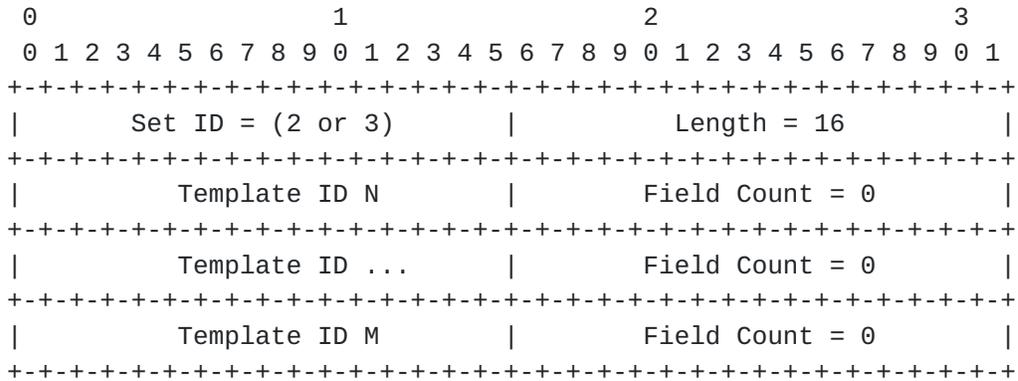
           Figure T: Template Withdrawal Message format

The Set ID field MUST contain the value 2 for Template Set withdrawal
and the value 3 for Options Template Set withdrawal. Multiple
Template IDs MAY be withdrawn with a single Template Withdrawal
Message: in that case, padding MAY be used.

The Template Withdraw Message withdraws the Template IDs for the
Observation Domain ID specified in the IPFIX Message header.

The Template Withdraw Message may be sent on any SCTP stream. The
Template Withdraw Message MUST be sent reliably, using SCTP ordered
delivery.

The Template Withdraw Message MUST NOT contain new Template or
Options Template Records.

If the measurement parameters change such that a new Template is
required, the Template MUST be withdrawn (using a Template Withdraw
Message and a new Template definition) or an unused Template ID MUST
be used. Examples of the measurement changes are: a new sampling
rate, a new flow expiration process, a new filtering definition, etc.

When the SCTP association shuts down or the Exporting Process
restarts, all Template assignments are lost and Template IDs MUST be
re-assigned.

   If the Metering Process restarts, the Exporting Process MUST either
   reuse the previously assigned Template ID for each Template, or it
   MUST withdraw the previously issued Template IDs by sending Template
   Withdraw Message(s) before reusing them.

   A Template Withdrawal Message to withdraw all Templates for the
   Observation Domain ID specified in the IPFIX Message header MAY be
   used. Its format is shown in figure U.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |              Set ID = 2        |            Length = 8         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |           Template ID = 2      |          Field Count = 0      |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
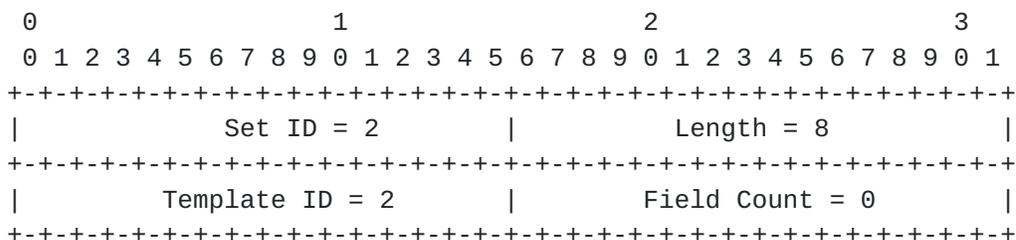
           Figure U: All Data Templates Withdrawal Message format

   A Template Withdrawal Message to withdraw all Options Templates for
   the Observation Domain ID specified in the IPFIX Message header MAY
   be used. Its format is shown in figure V.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |              Set ID = 3        |            Length = 8         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |           Template ID = 3      |          Field Count = 0      |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
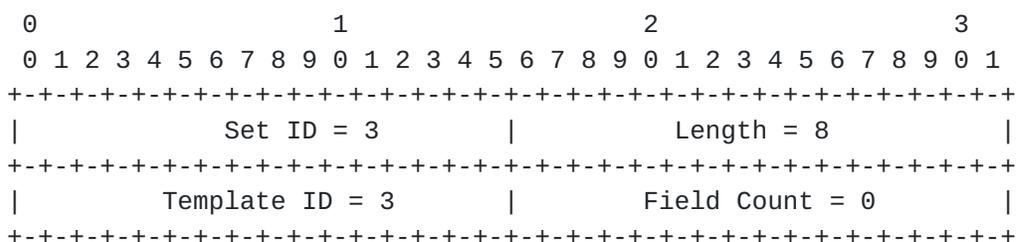
           Figure V: All Options Templates Withdrawal Message format

   When the SCTP association restarts, the Exporting Process MUST resend
   all the Template Records.

9.   The Collecting Process's Side

   This section describes the Collecting Process when using SCTP and
   PR-SCTP as the transport protocol. Any necessary changes to the
   Collecting Process specifically related to TCP or UDP transport
   protocols are specified in section 10.

   The Collecting Process SHOULD listen for a new association request
   from the Exporting Process. The Exporting Process will request a

   number of streams to use for export. An Exporting Process MAY
   request and support more than one stream per SCTP association.

   If the Collecting Process receives a malformed IPFIX Message, it
   MUST reset the SCTP association, discard the IPFIX Message, and
   SHOULD log the error. Note that non-zero Set padding does not
   constitute a malformed IPFIX Message.

   Template Sets and Option Template Sets are only sent once. The
   Collecting Process MUST store the Template Record information for
   the duration of the association so that it can interpret the
   corresponding Data Records that are received in subsequent Data
   Sets.

   Template IDs are unique per SCTP association and per Observation
   Domain. If the Collecting Process receives a Template which has
   already been received but which has not previously been withdrawn
   (i.e. a Template Record from the same Exporter Observation Domain
   with the same Template ID received on the SCTP association), then
   the Collecting Process MUST shutdown the association.

   When an SCTP association is closed, the Collecting Process MUST
   discard all Templates received over that association and stop
   decoding IPFIX Messages that use those Templates.

   The Collecting Process normally receives Template Records from the
   Exporting Process before receiving Data Records. The Data Records
   are then decoded and stored by the Collector. If the Template
   Records have not been received at the time Data Records are
   received, the Collecting Process MAY store the Data Records for a
   short period of time and decode them after the Template Records are
   received. A Collecting Process MUST NOT assume that the Data Set
   and the associated Template Set (or Options Template Set) are
   exported in the same IPFIX Message.

   The Collecting Process MUST note the Information Element identifier
   of any Information Element that it does not understand and MAY
   discard that Information Element from the Flow Record.

   The Collector MUST accept padding in Data Records and Template
   Records. The padding size is the Set Length minus the size of the
   Set Header (4 octets for the Set ID and the Set Length), modulo the
   Record size deduced from the Template Record.

   The IPFIX protocol has a Sequence Number field in the Export header
   which increases with the number of IPFIX Data Records in the IPFIX
   Message. A Collector may detect out of sequence, dropped, or
   duplicate IPFIX Messages by tracking the Sequence Number. A
   collector SHOULD provide a logging mechanism for tracking out of
   sequence IPFIX Messages. Such out of sequence IPFIX Messages may be
   due to Exporter resource exhaustion where it can not transmit
   messages at their creation rate, an Exporting Process reset,
   congestion on the network link between the Exporter and Collector,
   Collector resource exhaustion where it can not process the IPFIX
   Messages at their arrival rate, out of order packet reception,
   duplicate packet reception, or an attacker injecting false messages.

   If a Collecting Process receives a Template Withdraw Message, the
   Collecting Process MUST delete the corresponding Template Records
   associated with the specific SCTP association and specific
   Observation Domain, and stop decoding IPFIX Messages that use the
   withdrawn Templates.

   If the Collecting Process receives a Template Withdraw message for a
   Template Record it has not received before on this SCTP assocation,
   it MUST reset the SCTP association, discard the IPFIX Message, and
   SHOULD log the error as it does for malformed IPFIX Messages.

   A Collecting Process that receives IPFIX Messages from several
   Observation Domains on the same Transport Session MUST be aware that
   the uniqueness of the Template ID is not guaranteed across
   Observation Domains.

   The Collector MUST support the use of Templates containing multiple
   occurrences of the similar Information Elements.

10.   Transport Protocol

   The IPFIX Protocol Specification has been designed to be transport
   protocol independent. Note that the Exporter can export to multiple
   Collecting Processes, using independent transport protocols.

   The IPFIX Message Header 16-bit Length field limits the length of a
   IPFIX Message to 65535 octets including the header. A Collecting
   Process MUST be able to handle IPFIX Message lengths of up to 65535
   octets.

10.1   Transport Compliance and Transport Usage

   We need to differentiate between what must be implemented (so that
   operators can interoperably deploy compliant implementations from
   different vendors) and what should or could be used in various
   operational environments. We must also make sure that ALL
   implementations can operate in a congestion-aware and congestion
   avoidance mode.

   SCTP [RFC2960] and [RFC3309] using the PR-SCTP extension specified
   in [RFC3758] MUST be implemented by all compliant implementations.
   UDP [UDP] MAY also be implemented by compliant implementations. TCP
   [TCP] MAY also be implemented by compliant implementations.

   PR-SCTP SHOULD be used in deployments where Exporters 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 Exporters 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 Exporter and Collector MUST run
   in an environment where IPFIX traffic has been provisioned for or is
   contained through some other means.

10.2   SCTP

   This section describes how IPFIX can be transported over SCTP
   [RFC2960] and [RFC3309] using the PR-SCTP [RFC3758] extension.

10.2.1 Congestion Avoidance

   The SCTP transport protocol provides the required level of
   congestion avoidance by design.

   SCTP will detect congestion in the end-to-end path between
   the IPFIX Exporting Process and the IPFIX Collecting Process,
   and limit the transfer rate accordingly. When an IPFIX
   Exporting Process has records to export, but detects that
   transmission by SCTP is temporarily impossible, it can either
   wait until sending is possible again, or it can decide to drop the
   record. In the latter case, the dropped export data MUST

  be accounted for, so that the amount of dropped export data can be
  reported.

10.2.2 Reliability

  The SCTP transport protocol is by default reliable, but has the
  capability to deliver messages with partial reliability [RFC3758].

  Using reliable SCTP messages for the IPFIX export is not in itself a
  guarantee that all Data Records are delivered. If there is
  congestion on the link from the Exporting Process to the Collecting
  Process, or if a significant number of retransmissions are required,
  the send queues on the Exporting Process may fill up: the Exporting
  Process MAY either suspend export or discard the IPFIX Messages. If
  Data Records are discarded the IPFIX Sequence Numbers used for
  export MUST reflect the loss of data.

10.2.3 MTU

  SCTP provides the required IPFIX Message fragmentation service based
  on path MTU discovery.

10.2.4 Exporting Process

10.2.4.1 Association Establishment

  The IPFIX Exporting Process SHOULD initiate an SCTP association with
  the IPFIX Collecting Process. By default, the Collecting Process
  listens for connections on SCTP port 4739. By default, the
  Collecting Process listens for secure connections on SCTP port 4740
  (refer to the Security Considerations section). By default the
  Exporting Process tries to connect to one of these ports. It MUST
  be possible to configure both the Exporting and Collecting Processes
  to use a different SCTP port.

  The Exporting Process MAY establish more than one association
  (connection "bundle" in SCTP terminology) to the Collecting Process.

  An Exporting Process MAY support more than one active association
  to different Collecting Processes (including the case of different
  Collecting Processes on the same host).

10.2.4.2 Association Shutdown

   When an Exporting Process is shutdown, it SHOULD shutdown the SCTP
   association.

   When a Collecting Process no longer wants to receive IPFIX
   Messages, it SHOULD shutdown its end of the association. The
   Collecting Process SHOULD continue to receive and process
   IPFIX Messages until the Exporting Process has closed its end of the
   association.

   When a Collecting Process detects that the SCTP association has been
   abnormally terminated, it MUST continue to listen for a new
   association establishment.

   When an Exporting Process detects that the SCTP association to the
   Collecting Process is abnormally terminated, it SHOULD try to re-
   establish the association.

   Association timeouts SHOULD be configurable.

10.2.4.3 Stream

   An Exporting Process MAY request more than one SCTP stream per
   association. Each of these streams may be used for the transmission
   of IPFIX Messages containing Data Sets, Template Sets, and/or
   Options Template Sets.

   Depending on the requirements of the application, the Exporting
   Process may send Data Sets with full or partial reliability, using
   ordered or out-of-order delivery, over any SCTP stream established
   during SCTP Association setup.

   An IPFIX Exporting Process MAY use any PR-SCTP Service Definition as
   per Section 4 of the PR-SCTP [RFC3758] specification when using
   partial reliability to transmit IPFIX Messages containing only Data
   Sets.

   However, Exporting Processes SHOULD mark such IPFIX Messages for
   retransmission for as long as resource or other constraints allow.

10.2.4.4 Template Management

   When the transport protocol is SCTP the default Template Management
   described in Section 8 is used.

10.2.5 Collecting Process

  When the transport protocol is SCTP, the default Collector
  processing described in Section 9 is used.

10.2.6 Failover

  If the Collecting Process does not acknowledge the attempt by the
  Exporting Process to establish an association the Exporting Process
  should retry using the SCTP exponential backoff feature. The
  Exporter MAY log an alarm if the time to establish the association
  exceeds a specified threshold, configurable on the Exporter.

  If Collecting Process failover is supported by the Exporting Process
  a second SCTP association MAY be opened in advance.

10.3   UDP

  This section describes how IPFIX can be transported over UDP
  [UDP]

10.3.1 Congestion Avoidance

  UDP has no integral congestion avoidance mechanism. Its use
  over congestion sensitive network paths is therefore not
  recommended. UDP MAY be used in deployments where Exporters and
  Collectors always communicate over dedicated links that are not
  susceptible to congestion, i.e. over provisioned links compared to
  the maximum export rate from the Exporters.

10.3.2 Reliability

  UDP is not a reliable transport protocol, and cannot guarantee
  delivery of messages. IPFIX Messages sent from the Exporting
  Process to the Collecting Process using UDP may therefore be lost.
  UDP MUST NOT be used unless the application can tolerate some
  loss of IPFIX Messages.

  The Collecting Process SHOULD deduce the loss and reordering of
  IPFIX Data Records by looking at the discontinuities in the IPFIX
  Sequence Number. In the case of UDP, the IPFIX Sequence Number
  contains the total number of IPFIX Data Records sent for the UDP
  Transport Session, prior to the receipt of this IPFIX Message,
  modulo 2^{32}. A Collector SHOULD detect out of sequence, dropped, or
  duplicate IPFIX Messages by tracking the Sequence Number.

   Templates sent from the Exporting Process to the Collecting
   Process using UDP as a transport MUST be resent at regular
   intervals in case previous copies were lost.

10.3.3 MTU

   The maximum size of exported messages MUST be configured such that
   the total packet size does not exceed the path MTU. If the path MTU
   is unknown, a maximum packet size of 512 octets SHOULD be used.

10.3.4 Port Numbers

   By default, the Collecting Process listens on the UDP port 4739. By
   default, the Collecting Process listens for secure connections on UDP
   port 4740 (refer to the Security Considerations section). By default
   the Exporting Process tries to connect to one of these ports
      It MUST be possible to configure both the Exporting and Collecting
   Processes to use a different UDP port.

10.3.5 Exporting Process

   The Exporting Process MAY duplicate the IPFIX Message to the several
   Collecting Processes.

10.3.6 Template Management

   When IPFIX uses UDP as the transport protocol, Template Sets and
   Option Template Sets MUST be re-sent at regular intervals. The
   frequency of (Options) Template transmission MUST be configurable.
   The default value for the frequency of (Options) Template
   transmission is 10 minutes. The Exporting Process SHOULD transmit
   the Template Set and Options Template Set in advance of any Data
   Sets that use that (Options) Template ID, to help ensure that the
   Collector has the Template Record before receiving the first Data
   Record.

   In the event of configuration changes, the Exporting Process SHOULD
   send multiple copies of the new Template definitions, in different
   IPFIX Messages, at an accelerated rate. In such a case, it SHOULD
   transmit the changed Template Record(s) and Options Template
   Record(s), without any data, in advance to help ensure that the
   Collector will have the correct Template information before
   receiving the first data.

If the Option Template scope is defined in another Template, then
both Templates SHOULD be sent in the same IPFIX Message. For
example: if a Flow Key Option Template (see section 4.4) is sent in
an Option Template, then the associated Template SHOULD be sent in
the same IPFIX Message.

Following a configuration change that can modify the interpretation
of the Data Records (for example, a sampling rate change) a new
Template ID MUST be used and the old Template ID MUST NOT be reused
until its lifetime (see section 10.3.7) has expired.

If UDP is selected as the transport protocol, the Template Withdraw
Messages MUST not be used, as this method is inefficient due to the
unreliable nature of UDP.

10.3.7 Collecting Process

  The Collecting Process MUST associate a lifetime with each
  Template (or another of definition of an identifier considered unique
  within the Transport Session) received via UDP. Templates (and
  similar definitions) not refreshed by the Exporting Process within
  the lifetime are expired at the Collecting Process. If the Template
  (or other definition) is not refreshed before that lifetime has
  expired, the Collecting Process MUST discard that definition and any
  current and future associated Data Records. In which case, an alarm
  MUST be logged. The Collecting Process MUST NOT decode any further
  Data Records which are associated with the expired Template. If a
  Template is refreshed with a Template Record that differs from the
  previous received Template Record, the Collecting Process SHOULD log
  a warning and replace the previous received Template Record with the
  new one. The Template lifetime at the Collecting Process MUST be at
  least 3 times higher than the Template refresh timeout configured on
  the Exporting Process.

  Template IDs are unique per UDP session and per Observation Domain.
  At any given time the Collecting Process SHOULD maintain the
  following for all the current Template Records and Options Template
  Records: <IPFIX Device, Exporter source UDP port, Observation Domain
  ID, Template ID, Template Definition, Last Received>.

  The Collecting Process SHOULD accept Data Records without the
  associated Template Record (or other definitions) required to decode
  the Data Record. If the Template Records (or other definitions such
  as Common Properties) have not been received at the time Data Records
  are received, the Collecting Process SHOULD store the Data Records
  for a short period of time and decode them after the Template Records
  (or other definitions) are received. The short period of time MUST

   be lower than the lifetime of definitions associated with identifiers
   considered unique within the UDP session.

   If the Collecting Process receives a malformed IPFIX Message, it MUST
   discard the IPFIX Message, and SHOULD log the error.

10.3.8 Failover

   Because UDP is not a connection oriented protocol, the Exporting
   Process is unable to determine from the transport protocol that the
   Collecting Process is no longer able to receive the IFPIX Messages.
   Therefore, it can not invoke a failover mechanism. However, the
   Exporting Process MAY duplicate the IPFIX Message to several
   Collecting Processes.

10.4   TCP

   This section describes how IPFIX can be transported over TCP [TCP].

10.4.1 Connection Management

10.4.1.1 Connection Establishment

   The IPFIX Exporting Process initiates a TCP connection to the
   Collecting Process. By default, the Collecting Process listens for
   connections on TCP port 4739. By default, the Collecting Process
   listens for secure connections on TCP port 4740 (refer to the
   Security Considerations section). By default the Exporting Process
   tries to connect to one of these ports. It MUST be possible to
   configure both the Exporting Process and the Collecting Process to
   use a different TCP port.

   An Exporting Process MAY support more than one active connection to
   different Collecting Processes (including the case of different
   Collecting Processes on the same host).

   The Exporter MAY log an alarm if the time to establish the
   connection exceeds a specified threshold, configurable on the
   Exporter.

10.4.1.2 Graceful Connection Release

   When an Exporting Process is shutdown, it SHOULD shutdown the TCP
   connection.

   When a Collecting Process no longer wants to receive IPFIX Messages,
   it SHOULD close its end of the connection. The Collecting Process
   SHOULD continue to read IPFIX Messages until the Exporting Process
   has closed its end.

10.4.1.3 Restarting Interrupted Connections

   When a Collecting Process detects that the TCP connection to the
   Exporting Process has terminated abnormally, it MUST continue to
   listen for a new connection.

   When an Exporting Process detects that the TCP connection to the
   Collecting Process has terminated abnormally, it SHOULD try to re-
   establish the connection. Connection timeouts and retry schedules
   SHOULD be configurable. In the default configuration, an Exporting
   Process MUST NOT attempt to establish a connection more frequently
   than once per minute.

10.4.1.4 Failover

   If the Collecting Process does not acknowledge the attempt by the
   Exporting Process to establish a connection it will retry using the
   TCP exponential backoff feature.

   If Collecting Process failover is supported by the Exporting Process
   a second TCP connection MAY be opened in advance.

10.4.2 Data Transmission

   Once a TCP connection is established, the Exporting Process starts
   sending IPFIX Messages to the Collecting Process.

10.4.2.1 IPFIX Message Encoding

   IPFIX Messages are sent over the TCP connection without any special
   encoding. The Length field in the IPFIX Message header defines the
   end of each IPFIX Message and thus the start of the next IPFIX
   Message. This means that IPFIX Messages cannot be interleaved.

   In the case of TCP, the IPFIX Sequence Number contains the total
   number of IPFIX Data Records sent from this TCP connection, from the

   current Observation Domain by the Exporting Process, prior to the
   receipt of this IPFIX Message, modulo 2^32.

   If an Exporting Process exports data from multiple Observation
   Domains, it should be careful to choose IPFIX Message lengths
   appropriately to minimize head-of-line blocking between different
   Observation Domains. Multiple TCP connections MAY be used to avoid
   head-of-line between different Observation Domains.

10.4.2.2 Template Management

   For each Template, the Exporting Process MUST send the Template
   Record before exporting Data Records that refer to that Template.

   Template IDs are unique per TCP connection and per Observation
   Domain. A Collecting Process MUST record all Template and Options
   Template Records for the duration of the connection, as an Exporting
   Process is not required to re-export Template Records.

   When the TCP connection restarts, the Exporting Process MUST resend
   all the Template Records.

   When a TCP connection is closed, the Collecting Process MUST discard
   all Templates received over that connection and stop decoding IPFIX
   Messages that use those Templates.

   The Templates that are not used anymore SHOULD be deleted. Before
   reusing a Template ID, the Template MUST be deleted. In order to
   delete an allocated Template, the Template is withdrawn through the
   use of a Template Withdrawal Message over the TCP connection.

   If the Collecting Process receives a malformed IPFIX Message, it
   MUST reset the TCP connection, discard the IPFIX Message, and SHOULD
   log the error.

10.4.2.3 Congestion Handling and Reliability

   TCP ensures reliable delivery of data from the Exporting Process to
   the Collecting Process. TCP also controls the rate at which data
   can be sent from the Exporting Process to the Collecting Process,
   using a mechanism that takes into account both congestion in the
   network and the capabilities of the receiver.

Therefore an IPFIX Exporting Process may not be able to send IPFIX
Messages at the rate that the Metering Process generates it, either
because of congestion in the network or because the Collecting
Process cannot handle IPFIX Messages fast enough. As long as
congestion is transient, the Exporting Process can buffer IPFIX
Messages for transmission. But such buffering is necessarily
limited, both because of resource limitations and because of
timeliness requirements, so ongoing and/or severe congestion may
lead to a situation where the Exporting Process is blocked.

When an Exporting Process has Data Records to export but the
transmission buffer is full, and it wants to avoid blocking, it can
decide to drop some Data Records. The dropped Data Records MUST be
accounted for, so that the amount can later be exported.

When an Exporting Process finds that the rate at which records
should be exported is consistently higher than the rate at which TCP
sending permits, it should provide back pressure to the metering
processes. The metering process could then adapt by temporarily
reducing the amount of data it generates, for example using sampling
or aggregation.

10.4.3 Collecting Process

   The Collecting Process SHOULD listen for a new TCP connection from
   the Exporting Process.

   If the Collecting Process receives a malformed IPFIX Message, it
   MUST reset the TCP connection, discard the IPFIX Message, and SHOULD
   log the error. Note that non-zero Set padding does not constitute a
   malformed IPFIX Message.

   Template Sets and Option Template Sets are only sent once. The
   Collecting Process MUST store the Template Record information for
   the duration of the connection so that it can interpret the
   corresponding Data Records that are received in subsequent Data
   Sets.

   Template IDs are unique per TCP connection and per Observation
   Domain. If the Collecting Process receives a Template which has
   already been received but which has not previously been withdrawn
   (i.e. a Template Record from the same Exporter Observation Domain

with the same Template ID received on the TCP connection), then the
Collecting Process MUST shutdown the connection.

When a TCP connection is closed, the Collecting Process MUST discard
all Templates received over that connection and stop decoding IPFIX
Messages that use those Templates.

If a Collecting Process receives a Template Withdraw Message, the
Collecting Process MUST delete the corresponding Template Records
associated with the specific TCP connection and specific Observation
Domain, and stop decoding IPFIX Messages that use the withdrawn
Templates.

If the Collecting Process receives a Template Withdraw message for a
Template Record it has not received before on this TCP connection,
it MUST reset the TCP association, discard the IPFIX Message, and
SHOULD log the error as it does for malformed IPFIX Messages.

11.   Security Considerations

The security considerations for the IPFIX protocol have been derived
from an analysis of potential security threats, as discussed in the
security consideration section of IPFIX requirements [RFC3917]. The
requirements for IPFIX security are as follows:

  1. IPFIX must provide a mechanism to ensure the confidentiality of
  IPFIX data transferred from an Exporting Process to a Collecting
  Process, in order to prevent disclosure of flow data transported
  via IPFIX.

  2. IPFIX must provide a mechanism to ensure the integrity of IPFIX
  data transferred from an Exporting Process to a Collecting Process,
  in order to prevent the injection of incorrect data or control
  information (e.g. Templates) into an IPFIX Message stream.

  3. IPFIX must provide a mechanism to authenticate IPFIX Collecting
  and Exporting Processes, to prevent the collection of data from an
  unauthorized Exporting Process, or the export of data to an
  unauthorized Collecting Process

Because IPFIX can be used to collect information for network
forensics and billing purposes, attacks designed to confuse, disable,
or take information from an IPFIX collection system may be seen as a
prime objective during a sophisticated network attack.

An attacker in a position to inject false messages into an IPFIX
Message stream can either affect the application using IPFIX (by

falsifying data), or the IPFIX Collecting Process itself (by
modifying or revoking Templates, or changing options); for this
reason, IPFIX Message integrity is important.

The IPFIX Messages themselves may also contain information of value
to an attacker, including information about the configuration of the
network as well as end-user traffic and payload data, so care must be
taken to confine their visibility to authorized users. When an
Information Element containing end-user payload information is
exported, it SHOULD be transmitted to the Collecting Process using a
means that secures its contents against eavesdropping. Suitable
mechanisms include the use of either a direct point-to-point
connection or the use of an encryption mechanism. It is the
responsibility of the Collecting Process to provide a satisfactory
degree of security for this collected data, including, if necessary,
anonymization of any reported data.

11.1   Applicability of TLS and DTLS

TLS [RFC4346] and DTLS [RFC4347] were designed to provide the
confidentiality, integrity, and authentication assurances required by
the IPFIX protocol, without the need for pre-shared keys.

With the mandatory SCTP and PR-SCTP transport protocols for IPFIX,
DTLS [RFC4347] MUST be implemented. If UDP is selected as the IPFIX
transport protocol, DTLS [RFC4347] MUST be implemented. If TCP is
selected as the IPFIX transport protocol, TLS [RFC4346] MUST be
implemented.

Note that DTLS is selected as the security mechanism for SCTP and
PR-SCTP. Though TLS bindings to SCTP are defined in [RFC3436], they
require all communication to be over reliable, bi-directional
streams, and require one TLS connection per stream. This arrangement
is not compatible with the rationale behind the choice of SCTP as an
IPFIX transport protocol.

Note that using DTLS [RFC4347] has a vulnerability, i.e. a true man
in the middle may attempt to take data out of an association and fool
the sender into thinking that the data was actually received by the
peer. In generic TLS for SCTP (and/or TCP) this is not possible.
This means that the removal of a message may become hidden from the
sender or receiver. Another vulnerability of using PR-SCTP with DTLS
is that someone could inject SCTP control information to shut down
the SCTP association, effectively generating a loss of IPFIX Messages
if those are buffered outside of the SCTP association. In the future,
techniques such as [dtls-for-sctp] could be used to overcome these
vulnerabilities.

   When using DTLS over SCTP, the Exporting Process MUST ensure that
   each IPFIX Message is sent over the same SCTP stream that would be
   used when sending the same IPFIX Message directly over SCTP. Note
   that DTLS may send its own control messages on stream 0 with full
   reliability; however, this will not interfere with the processing of
   stream 0 IPFIX Messages at the Collecting Process, because DTLS
   consumes its own control messages before passing IPFIX Messages up to
   the application layer.

11.2   Usage

   The IPFIX Exporting Process initiates the communication to the IPFIX
   Collecting Process, and acts as a TLS or DTLS client according to
   [RFC4346] and [RFC4347], while the IPFIX Collecting Process acts as a
   TLS or DTLS server. The DTLS client opens a secure connection on the
   SCTP port 4740 of the DTLS server if SCTP or PR-SCTP is selected as
   the transport protocol. The TLS client opens a secure connection on
   the TCP port 4740 of the TLS server if TCP is selected as the
   transport protocol. The DTLS client opens a secure connection on the
   UDP port 4740 of the DTLS server if UDP is selected as the transport
   protocol.

11.3   Authentication

   IPFIX Exporting Processes and IPFIX Collecting Processes are
   identified by the fully-qualified domain name of the interface on
   which IPFIX Messages are sent or received, for purposes of X.509
   client and server certificates as in [RFC3280].

   To prevent man-in-the-middle attacks from impostor Exporting or
   Collecting Processes, the acceptance of data from an unauthorized
   Exporting Process, or the export of data to an unauthorized
   Collecting Process, strong mutual authentication via asymmetric keys
   MUST be used for both TLS and DTLS. Each of the IPFIX Exporting and
   Collecting Processes MUST verify the identity of its peer against its
   authorized certificates, and MUST verify that the peer's certificate
   matches its fully-qualified domain name, or, in the case of SCTP, the
   fully-qualified domain name of one of its endpoints.

   The fully-qualified domain name used to identify an IPFIX Collecting
   Process or Exporting Process may be stored either in a subjectAltName
   extension of type dNSName, or in the most specific Common Name field
   of the Subject field of the X.509 certificate. If both are present,
   the subjectAltName extension is given preference.

11.4   Protection against DoS attacks

   An attacker may mount a denial of service attack against an IPFIX
   collection system either directly, by sending large amounts of
   traffic to a Collecting Process, or indirectly, by generating large
   amounts of traffic to be measured by a Metering Process.

   Direct denial of service attacks can also involve state exhaustion,
   whether at the transport layer (e.g., by creating a large number of
   pending connections), or within the IPFIX Collecting Process itself
   (e.g., by sending Flow Records pending Template or Scope information,
   a large amount of Options Template Records, etc.)

   SCTP mandates a cookie exchange mechanism designed to defend against
   SCTP state exhaustion denial of service attacks. Similarly, TCP
   provides the "SYN cookie" mechanism to mitigate state exhaustion; SYN
   cookies SHOULD be used by any Collecting Process accepting TCP
   connections. DTLS also provides cookie exchange to protect against
   DTLS server state exhaustion.

   The reader should note that there is no way to prevent fake IPFIX
   Message processing (and state creation) for UDP & SCTP communication.
   The use of TLS and DTLS can obviously prevent the creation of fake
   states but they are themselves prone to state exhaustion attacks.
   Therefore, Collector rate limiting SHOULD be used to protect TLS &
   DTLS (like limiting the number of new TLS or DTLS session per second
   to a sensible number).

   IPFIX state exhaustion attacks can be mitigated by limiting the rate
   at which new connections or associations will be opened by the
   Collecting Process, the rate at which IPFIX Messages will be accepted
   by the Collecting Process, and adaptively limiting the amount of
   state kept, particularly records waiting on Templates. These rate
   and state limits MAY be provided by a Collecting Process; if
   provided, the limits SHOULD be user-configurable.

   Additionally, an IPFIX Collecting Process can eliminate the risk of
   state exhaustion attacks from untrusted nodes by requiring TLS or
   DTLS mutual authentication, causing the Collecting Process to accept
   IPFIX Messages only from trusted sources.

   With respect to indirect denial of service, the behavior of IPFIX
   under overload conditions depends on the transport protocol in use.
   For IPFIX over TCP, TCP congestion control would cause the flow of
   IPFIX Messages to back off and eventually stall, blinding the IPFIX
   system. PR-SCTP improves upon this situation somewhat, as some IPFIX
   Messages would continue to be received by the Collecting Process due
   to the avoidance of head-of-line blocking by SCTP's multiple streams
   and partial reliability features, possibly affording some visibility

of the attack. The situation is similar with UDP, as some datagrams
may continue to be received at the Collecting Process, effectively
applying sampling to the IPFIX Message stream, implying that some
forensics may be left.

To minimize IPFIX Message loss under overload conditions, some
mechanism for service differentiation could be used to prioritize
IPFIX traffic over other traffic on the same link. Alternatively,
IPFIX Messages can be transported over a dedicated network. In this
case, care must be taken to ensure that the dedicated network can
handle the expected peak IPFIX Message traffic.

11.5   When DTLS or TLS is not an option

The use of DTLS or TLS might not be possible in some cases due to
performance issues or other operational concerns.

Without TLS or DTLS mutual authentication, IPFIX Exporting Processes
and Collecting Processes can fall back on using IP source addresses
to authenticate their peers. A policy of allocating Exporting
Process and Collecting Process IP addresses from specified address
ranges, and using ingress filtering to prevent spoofing, can improve
the usefulness of this approach. Again, completely segregating IPFIX
traffic on a dedicated network, where possible, can improve security
even further. In any case, the use of open Collecting Processes
(those which will accept IPFIX Messages from any Exporting Process
regardless of IP address or identity) is discouraged.

Modern TCP and SCTP implementations are resistent to blind insertion
attacks (see [RFC1948], [RFC2960]); however, UDP offers no such
protection. For this reason, IPFIX Message traffic transported via
UDP and not secured via DTLS SHOULD be protected via segregation to a
dedicated network.

11.6   Logging an IPFIX Attack

IPFIX Collecting Processes MUST detect potential IPFIX Message
insertion or loss conditions by tracking the IPFIX Sequence Number,
and SHOULD provide a logging mechanism for reporting out of sequence
messages. Note that an attacker may be able to exploit the handling
of out of sequence messages at the Collecting Process, so care should
be taken in handling these conditions. For example, a Collecting
Process that simply resets the expected Sequence Number upon receipt
of a later Sequence Number could be temporarily blinded by deliberate
injection of later Sequence Numbers.

IPFIX Exporting and Collecting Processes SHOULD log any connection

   attempt that fails due to authentication failure, whether due to
   being presented an unauthorized or mismatched certificate during TLS
   or DTLS mutual authentication, or due to a connection attempt from an
   unauthorized IP address when TLS or DTLS are not in use.

   IPFIX Exporting and Collecting Processes SHOULD detect and log any
   SCTP association reset or TCP connection reset.

 11.7   Securing the Collector

   The security of the Collector and its implementation is important to
   achieve overall security. However, it is outside the scope of this
   document.

 12.   IANA Considerations

   IPFIX Messages use two fields with assigned values. These are the
   IPFIX Version Number, indicating which version of the IPFIX Protocol
   was used to export an IPFIX Message, and the IPFIX Set ID,
   indicating the type for each set of information within an IPFIX
   Message.

   The IPFIX Version Number value of 10 is reserved for the IPFIX
   Protocol specified in this document. Set ID values of 0 and 1 are
   not used for historical reasons [RFC3954]. The Set ID value of 2 is
   reserved for the Template Set. The Set ID value of 3 is reserved
   for the Option Template Set. All other Set ID values from 4 to 255
   are reserved for future use. Set ID values above 255 are used for
   Data Sets.

   New assignments in either IPFIX Version Number or IPFIX Set ID
   assignments require a Standards Action [RFC2434], i.e. they are to
   be made via Standards Track RFCs approved by the IESG.

 13.   Appendix A

   This appendix, which is a not a normative reference, contains IPFIX
   encoding examples.

   Let's consider the example of an IPFIX Message composed of a
   Template Set, a Data Set (which contains three Data Records), an
   Options Template Set and a Data Set (which contains 2 Data Records
   related to the previous Options Template Record).

   IPFIX Message:

   +--------+--------------------------------------. . .
   | | +--------------+ +-----------------+
   |Message | | Template | | Data |
   | Header | | Set | | Set | . . .
   | | | (1 Template) | | (3 Data Records) |
   | | +--------------+ +-----------------+
   +--------+--------------------------------------. . .

       . . .---+
             +-----------------+ +-----------------+ |
             | Options | | Data | |
       . . . | Template Set | | Set | |
             | (1 Template) | | (2 Data Records) | |
             +-----------------+ +-----------------+ |
       . . .---+
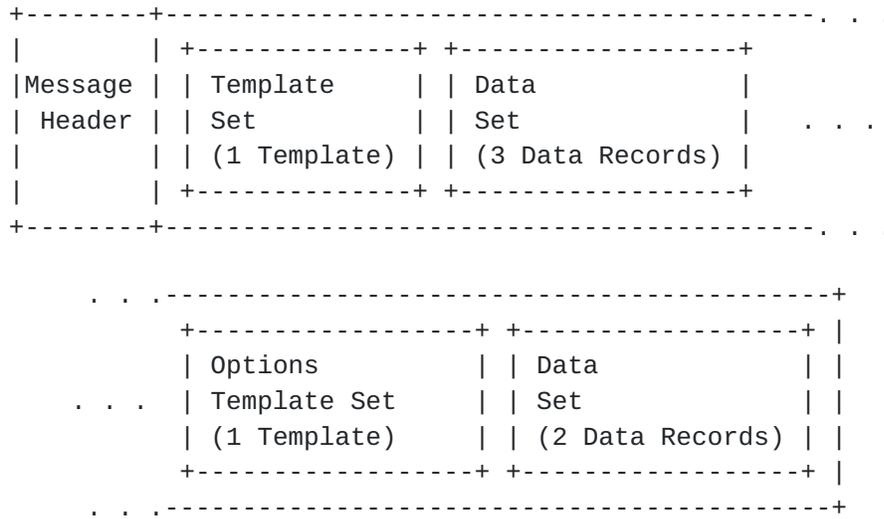
13.1   Message Header Example

   The Message Header is composed of:
    0   1   2   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+
   | Version = 0x000a | Length = 152 |
   +-+
   | Export Time |
   +-+
   | Sequence Number |
   +-+
   | Observation Domain ID |
   +-+
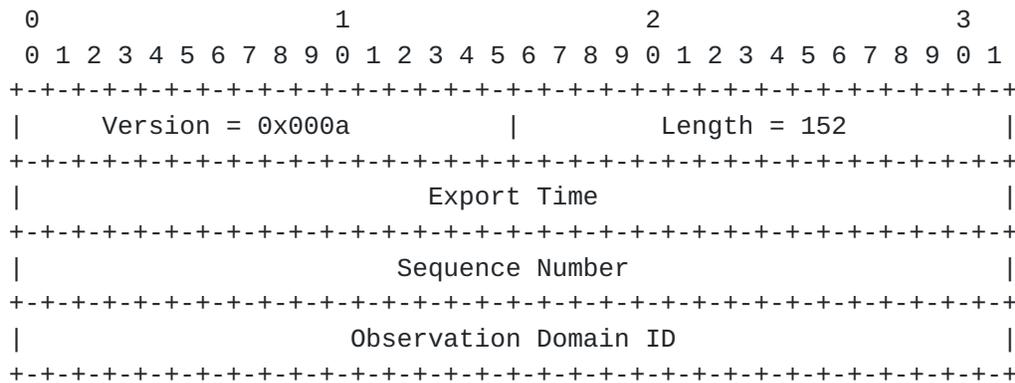
13.2   Template Set Examples

13.2.1 Template Set using IETF specified Information Elements

   We want to report the following Information Elements:

   - The IPv4 source IP address: sourceIPv4Address in [IPFIX-INFO],
   with a length of 4 octets
   - The IPv4 destination IP address: destinationIPv4Address in [IPFIX-
   INFO], with a length of 4 octets

   - The next-hop IP address (IPv4): ipNextHopIPv4Address in [IPFIX-
   INFO], with a length of 4 octets

   - The number of packets of the Flow: inPacketDeltaCount in [IPFIX-
   INFO], with a length of 4 octets

   - The number of octets of the Flow: inOctetDeltaCount in [IPFIX-
   INFO], with a length of 4 octets

   Therefore, the Template Set will be composed of the following:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Set ID = 2            |        Length = 28 octets      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Template ID 256        |        Field Count = 5         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|    sourceIPv4Address = 8    |        Field Length = 4        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0| destinationIPv4Address = 12 |        Field Length = 4        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|  ipNextHopIPv4Address = 15  |        Field Length = 4        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|   inPacketDeltaCount = 2    |        Field Length = 4        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|   inOctetDeltaCount =  1    |        Field Length = 4        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

13.2.2 Template Set using enterprise-specific Information Elements

   We want to report the following Information Elements:

   - The IPv4 source IP address: sourceIPv4Address in [IPFIX-INFO],
     with a length of 4 octets

   - The IPv4 destination IP address: destinationIPv4Address in
     [IPFIX-INFO], with a length of 4 octets

   - An enterprise-specific Information Element representing
     proprietary information, with a type of 15 and a length of 4

      - The number of packets of the Flow: inPacketDeltaCount in
        [IPFIX-INFO], with a length of 4 octets

      - The number of octets of the Flow: inOctetDeltaCount in
        [IPFIX-INFO], with a length of 4 octets

   Therefore, the Template Set will be composed of the following:

```
      0                   1                   2                   3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |           Set ID = 2          |        Length = 32 octets     |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |         Template ID 257       |        Field Count = 5        |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |0|    sourceIPv4Address = 8    |        Field Length = 4       |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |0| destinationIPv4Address = 12 |        Field Length = 4       |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |1| Information Element Id. = 15|        Field Length = 4       |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                         Enterprise number                     |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |0|    inPacketDeltaCount = 2   |        Field Length = 4       |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |0|    inOctetDeltaCount = 1    |        Field Length = 4       |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

13.3   Data Set Example

   In this example, we report the following three Flow Records:

```
   Src IP addr. | Dst IP addr.  | Next Hop addr. | Packet | Octets
                |               |                | Number | Number
   -----------------------------------------------------------------
   192.0.2.12   | 192.0.2.254   | 192.0.2.1      | 5009   | 5344385
   192.0.2.27   | 192.0.2.23    | 192.0.2.2      | 748    | 388934
   192.0.2.56   | 192.0.2.65    | 192.0.2.3      | 5      | 6534
```

```
      0                   1                   2                   3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |          Set ID = 256         |          Length = 64          |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
       |                         192.0.2.12                            |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                         192.0.2.254                           |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                         192.0.2.1                             |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                           5009                                |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                         5344385                               |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                         192.0.2.27                            |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                         192.0.2.23                            |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                         192.0.2.2                             |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                           748                                 |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                         388934                                |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                         192.0.2.56                            |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                         192.0.2.65                            |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                         192.0.2.3                             |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                           5                                   |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                          6534                                 |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

   Note that padding is not necessary in this example.

13.4   Options Template Set Examples

13.4.1 Options Template Set using IETF specified Information Elements

   Per line card (the router being composed of two line cards), we want
   to report the following Information Elements:

      - Total number of IPFIX Messages: exportedPacketCount
        [IPFIX-INFO], with a length of 2 octets

      - Total number of exported Flows: exportedFlowCount [IPFIX-INFO],
        with a length of 2 octets

   The line card, which is represented by the lineCardId Information
   Element [IPFIX-INFO], is used as the Scope Field.

   Therefore, the Options Template Set will be:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Set ID = 3           |           Length = 24         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Template ID 258        |        Field Count = 3        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Scope Field Count = 1     |0|    lineCardId = 141         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Scope 1 Field Length = 4    |0| exportedPacketCount = 41    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       Field Length = 2        |0|   exportedFlowCount = 42    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       Field Length = 2        |           Padding             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
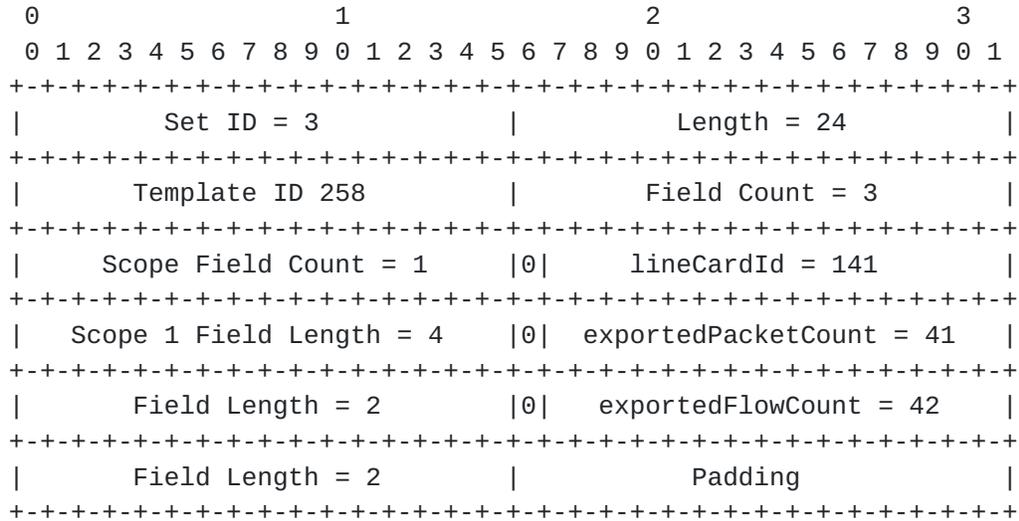

13.4.2 Options Template Set using enterprise-specific Information
          Elements
   Per line card (the router being composed of two line cards), we want
   to report the following Information Elements:

     - Total number of IPFIX Messages: exportedPacketCount
       [IPFIX-INFO], with a length of 2 octets

     - An enterprise-specific number of exported Flows,
       with a type of 42 and a length of 4 octets

   The line card, which is represented by the lineCardId Information
   Element [IPFIX-INFO], is used as the Scope Field.

   The format of the Options Template Set is as follows:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
```

```
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |          Set ID = 3           |            Length = 28        |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |         Template ID 259       |        Field Count = 3        |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |      Scope Field Count = 1    |0|     lineCardId = 141         |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |    Scope 1 Field Length = 4   |0| exportedPacketCount = 41    |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |         Field Length = 2      |1|Information Element Id. = 42 |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |         Field Length = 4      |         Enterprise number   ...
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       ...       Enterprise number     |           Padding             |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

13.4.3 Options Template Set using an enterprise-specific scope

   In this example, we want to export the same information as in the
   example in section 13.4.1:

      - Total number of IPFIX Messages: exportedPacketCount
         [IPFIX-INFO], with a length of 2 octets

      - Total number of exported Flows: exportedFlowCount
         [IPFIX-INFO], with a length of 2 octets

   But this time, the information pertains to a proprietary scope,
   identified by enterprise-specific Information Element number 123.

   The format of the Options Template Set is now as follows:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |          Set ID = 3           |            Length = 28         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |         Template ID 260       |        Field Count = 3         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |      Scope Field Count = 1    |1|Scope 1 Infor. El. Id. = 123 |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |     Scope 1 Field Length = 4  |        Enterprise Number    ...
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
 ...          Enterprise Number       |0|  exportedPacketCount = 41   |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |        Field Length = 2         |0|   exportedFlowCount = 42     |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |        Field Length = 2         |              Padding          |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

13.4.4 Data Set using an enterprise-specific scope

   In this example, we report the following two Data Records:

```
   Line Card ID              | IPFIX Message  | Exported Flow Records
   ---------------------------------------------------------------------
   Line Card 1 (lineCardId=1) | 345           | 10201
   Line Card 2 (lineCardId=2) | 690           | 20402
```

```
      0                   1                   2                   3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |      Set ID = 260                |           Length = 20         |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                               1                                 |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |              345                |              10201            |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                               2                                 |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |              690                |              20402            |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

13.5   Variable length Information Element examples

13.5.1 Example of Variable Length Information Element with Length
   inferior to 255 octets

```
      0                   1                   2                   3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |      5       |          5 octet Information Element           |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                     |
     +-+-+-+-+-+-+-+-+-+-+-+-+
```

13.5.2 Example of Variable Length Information Element with Length 255
        to 65535 octets

```
   0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |      255      |              1000             |    IE ...     |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                   1000 octet Information Element              |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   :                            ...                                :
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                          ... IE          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
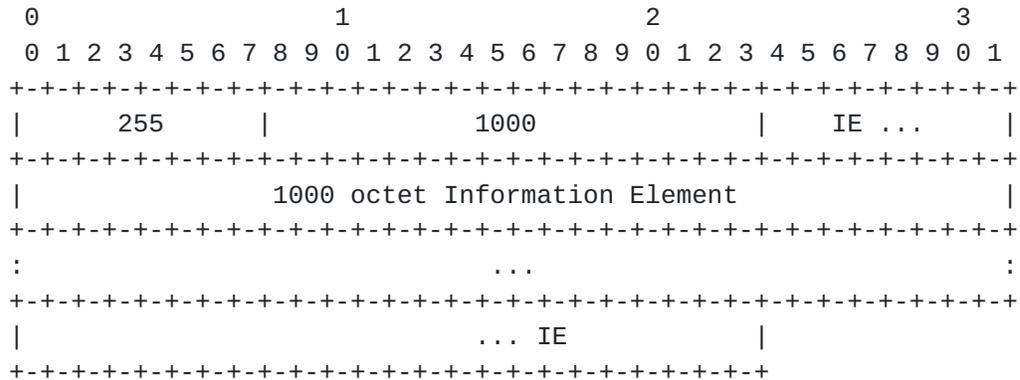
14.   References

14.1   Normative References

  [IPFIX-INFO] Quittek, J., Bryant S., Claise, B., Meyer, J.
  "Information Model for IP Flow Information Export" draft-ietf-ipfix-
  info-15, February 2007

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

  [TCP]   "TRANSMISSION CONTROL PROTOCOL DARPA INTERNET PROGRAM PROTOCOL
  SPECIFICATION" RFC 793, September 1981

  [RFC1305] Mills, D., "Network Time Protocol (Version 3)
  Specification, Implementation and Analysis", RFC 1305, May 1992

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

  [RFC2434] Alvestrand, H. and T. Narten, "Guidelines for Writing an
  IANA Considerations Section in RFCs", RFC 2434, October 1998.

  [RFC2960] Stewart, R. (ed.) "Stream Control Transmission Protocol",
  RFC 2960, October 2000

  [RFC3309] Steon, J., Stewart, R., Otis, D. "Stream Control
  Transmission Protocol (SCTP) Checksum Change", RFC 3309, September
  2002

[RFC3280] Housley, R., Polk, W., Ford, W., Solo, D. "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002

[RFC3436] Rescorla, E., Tuexen, M., "Transport Layer Security over Stream Control Transmission Protocol", RFC 3436, December 2002

[RFC3758] Stewart, R., Ramalho, M., Xie, Q., Tuexen, M., Conrad, P. "Stream Control Transmission Protocol (SCTP) Partial Reliability Extension", RFC 3758, May 2004

[RFC4346] Dierks, T. and C. Allen, "The TLS Protocol Version 1.1", RFC 4346, April 2006.

[RFC4347] Rescola, E., Modadugy, N. "Datagram Transport Layer Security", RFC 4347, April 2006

14.2   Informative References

[IPFIX-ARCH] Sadasivan, G., Brownlee, N., Claise, B., Quittek, J., "Architecture Model for IP Flow Information Export" draft-ietf-ipfix-architecture-12, September 2006

[IPFIX-AS] Zseby, T., Boschi, E., Brownlee, N., Claise, B., "IPFIX Applicability", draft-ietf-ipfix-as-12.txt, June 2007

[PEN] IANA Private Enterprise Numbers registry http://www.iana.org/assignments/enterprise-numbers

[RFC1948] Bellovin, S., "Defending Against Sequence Number Attacks", RFC 1948, May 1996

[RFC2579] McCloghrie, K., et al "Textual Conventions for SMIv2", RFC 2579, April 1999

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

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., Jacobson, V., "RTP: A Transport Protocol for Real-Time Applications ", RFC 3550, July 2003

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

[IEEE.754.1985] Institute of Electrical and Electronics Engineers, "Standard for Binary Floating-Point Arithmetic", IEEE Standard 754, August 1985

   [dtls-for-sctp], Tuexen, M., Hohendorf, C., Rescola, E., "Datagram
   Transport Layer Security for Stream Control Transmission Protocol",
   draft-tuexen-dtls-for-sctp-01.txt, October 2006

15.   Acknowledgments

   We would like to thank the following persons: Ganesh Sadasivan for
   his significant contribution during the initial phases of the
   protocol specification, Juergen Quittek for the coordination job
   within IPFIX and PSAMP; Nevil Brownlee, Dave Plonka, Paul Aitken,
   and Andrew Johnson for the thorough reviews; Randall Stewart and
   Peter Lei for their SCTP expertise and contributions; Martin
   Djernaes for the first essay on the SCTP section; Michael Behringer
   and Eric Vyncke for their advices and knowledge in security; Michael
   Tuexen for his help regarding the DTLS section; Elisa Boschi for her
   contribution regarding the improvement of SCTP sections; Mark
   Fullmer, Sebastian Zander, Jeff Meyer, Maurizio Molina, Carter
   Bullard, Tal Givoly, Lutz Mark, David Moore, Robert Lowe, Paul
   Calato, and many more, for the technical review and feedback.

Authors' Addresses

   Benoit Claise
   Cisco Systems
   De Kleetlaan 6a b1
   1831 Diegem
   Belgium
   Phone: +32 2 704 5622
   E-mail: bclaise@cisco.com

   Stewart Bryant
   Cisco Systems, Inc.
   250, Longwater,
   Green Park,
   Reading, RG2 6GB,
   United Kingdom
   Phone: +44 (0)20 8824-8828
   Email: stbryant@cisco.com

   Simon Leinen
   SWITCH
   Limmatquai 138
   P.O. Box

CH-8021 Zurich
Switzerland
Phone: +41 1 268 1536
EMail: simon@switch.ch

Thomas Dietz
NEC Europte Ltd.
Network Laboratories
Kurfuersten-Anlage 36
69115 Heidelberg
Germany
Phone: +49 6221 90511-28
Email: dietz@netlab.nec.de

Brian H. Trammell
CERT Network Situational Awareness
Software Engineering Institute
4500 Fifth Avenue
Pittsburgh, PA 15213
United States
Phone: +1 412 268 9748
Email: bht@cert.org

16.   Intellectual Property Statement

17.   Copyright Statement

   Copyright (C) The IETF Trust (2007).

   This document is subject to the rights, licenses and
   restrictions contained in BCP 78, and except as set forth
   therein, the authors retain all their rights.

18.   Disclaimer

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