IPFIX Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 28, 2011

A. Akhter
Cisco Systems

October 25, 2010

Information Elements for Flow Performance Measurement
draft-akhter-ipfix-perfmon-01.txt

Abstract

There is a need to be able to quantify and report the performance of network applications and the network service in handling user data. This performance data provides information essential in validating service level agreements, fault isolation as well as early warnings of greater problems. This document describes IPFIX Information Elements related to performance measurement of network based applications. In addition, to the performance information several non-metric information elements are also included to provide greater context to the reports. The measurements use audio/video applications as a base but are not restricted to these class of applications.

Status of this Memo

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

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

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

This Internet-Draft will expire on April 28, 2011.

Table of Contents

1. Introduction

Today's networks support a multitude of highly demanding and sensitive network applications. Network issues are readily apparent by the users of these applications due to the sensitivity of these applications to impaired network conditions. Examples of these network applications include applications making use of IP based audio, video, database transactions, virtual desktop interface (VDI), online gaming, cloud services and many more. In some cases the impaired application translates directly to loss of revenue. In other cases, there may be regulatory or contractual service level agreements that motivate the network operator. Due to the sensitive of these types of applications to impaired service it leaves a poor impression of the service on the user-- regardless of the actual performance of the network itself. In the case of an actual problem within the network service, monitoring the performance may yield a early indicator of a much more serious problem.

Due to the demanding and sensitive nature of these applications, network operators have tried to engineer their networks in an attempt to wring better and differentiated performance. However, that same differentiated design prevents network operators from extrapolating observational data from one application to another, or from one set of synthetic (active test) test traffic to actual application performance.

Performance measurements on user data provide greater visibility not only into the quality of experience of the end users but also visibility into network health. With regards to network health, as flow performance is being measured, there will be visibility into the end to end performance which means that not only visibility into local network health, but also viability into remote network health. If these measurements are made at multiple points within the network (or between the network and end device) then there is not only identification that there might be an issue, but a span of area can be established where the issue might be. The resolution of the fault increases with the number of measurement points along the flow path.

The IP Flow Information Export Protocol (IPFIX) [RFC5101] provides new levels of flexibility in reporting from measurement points across the life cycle of a network based application. IPFIX can provide granular results in terms of flow specificity as well as time granularity. At the same time, IPFIX allows for summarization of data along different types of boundaries for operators that are unconcerned about specific sessions but about health of a service or a portion of the network.

Where possible, an attempt has been made to make use of existing

definitions of metrics ([RFC4710]) and if needed, clarify and expand on them to widen their usage with additional applications. The methodology described in [I-D.ietf-pmol-sip-perf-metrics] is used to describe the methodology of measurement. As this document also covers the reporting of these metrics via IPFIX, consideration is taken with mapping the metric's capabilities and context with the IPFIX information and data representation model. The guidelines outlined in [I-D.trammell-ipfix-ie-doctors] are used to ensure proper IPFIX information element definition.

There has been related work in this area such as [RFC2321]. [I-D.huici-ipfix-sipfix], and [VoIP-monitor]. This document is also an attempt to generalize as well as standardize the reporting formats and measurement methodology.

2. Terminology

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

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].

In addition, the information element definitions use the following terms:

Name: Name of the information element per the IPFIX rules defined in Section 2.3 of [RFC5102]

Description: Short description of what the information element is trying to convey.

Observation Point: Where the measurement is meant to be performed. Either at an intermediate point (for example, a router) or end system.

Element Data Type: The IPFIX informationElementDataTypeas defined in Section 3.1 of [RFC5610]

Element Semantics: The IPFIX informationElementSemantics as defined in section Section 3.6 of [RFC5610]

Element Units: The IPFIX informationElementUnits as defined in section Section 3.7 of [RFC5610]

Element Range Begin: The IPFIX informationElementRangeBegin as defined in section Section 3.7 of [RFC5610]

Element Range End: The IPFIX informationElementRangeEnd as defined in section Section 3.7 of [RFC5610]

Element Id: The IPFIX global unique element ID as defined in Section 3.2 of [RFC5101]

Status: The status of the specification of this IPFIX Information Element.

Use and Applications An explanation of how this particular information element would be used.

Calculation Method: In the case of metrics, this section describes how the metric is calculated, as well as any special conditions.

Units of Measurement: In the case of metrics, what are the units of measurement. The text here is expected to be wider and more descriptive than in the IPFIX Element Units section.

Measurement Timing: Discussion on the acceptable range of timing and sampling intervals.

3. General Usage

3.1. Quality of Service (QoS) Monitoring

The network operator needs to be able to gauge the end user’s satisfaction with the network service. While there are many components of the satisfaction such as pricing, packaging, offering, etc., a major component of satisfaction is delivering a consistent service. The user builds trust on this consistency of the network service and is then to be able to run network applications-- which is of course the end goal. Without the ability to deliver a consistent service for end user network applications network operator will be left dealing with price sensitive disgruntled users with very low expectations (if they don’t have choice of operator) or abandonment (if they have choice).

3.2. Service Level Agreemnt (SLA) Validation

Similar to QoS and QoE validation, there might be contractual or regulatory requirements that need to be met by the network operator. Monitoring the performance of the flows allows the application operator, network operator as well as the end user to validate of the target service is being delivered. While there is quite a diversity in the codification of network SLAs they may eventually involve some measurement of network uptime, end to end latency, end to end jitter and perhaps service response time. In the case violation of the SLA, the start and end times, nature and network scope of the violation needs to be captured to allow for the most accurate settling of the SLA.

3.3. Fault Isolation and Troubleshooting

It has been generally easier to troubleshoot and fix problems that are binary in nature: it either works or does not work. The host is pingable or not pingable. However, the much more difficult to resolve issues that are transitory in nature, move from location to location, more complicated that simple ICMP reachability and many times unverifiable reports by the users themselves. It is these intermittent and seemingly inconsistent network impairments that performance metrics can be extremely helpful with. Just the basic timely detection that there is a problem (or an impending problem) can give the provider the confidence that there is a real problem that needs to be resolved. The next step would be to assist the operator in a speedy resolution by providing information regarding the network location and nature of the problem.

4. New Information Elements

The information elements are organized into two main groups:

Transport Layer: Metrics that might be calculated from observations at higher layers but essentially provide information about the network transport of user date. For example, the metrics related to packet loss, latency and jitter would be defined here.

User and Application Layer: Metrics that are might be affected by the network indirectly, but are ultimately related to user, end-system and session states. For example, session setup time, transaction rate and session duration would be defined here.

Contextual Elements Information elements that provide further context to the metrics. For example, media type, codec type, and type of application would be defined here.

4.1. Transport Layer

4.1.1. perfPacketLoss

Name: perfPacketLoss

Description: The packet loss metric reports the number of individual packets that were lost in the reporting interval.

Observation Point: The observation can be made anywhere along the media path or on the endpoints them selves. The observation is only relevant in a unidirectional sense.

Element Data Type: unsigned32

Element Semantics: deltaCounter

Element Units: packets

Element Range Begin: 0

Element Range End: 0xFFFFFFFE

Element Id: TBDperfPacketLoss

Status: current

Use and Applications The packet loss metric can be used to determine if there is a network impairment that is causing packet loss upstream of the measurement point. When there are observation points on either side of the impairment location it is possible to locate the impairment. With the location information the operator can is able to perform quicker fault-isolation as well as shorten time to resolution.

Calculation Method: This metric requires that each IP packet be individually marked with a monotonically incrementing sequence number. A number of encapsulations support this type of sequencing: IPSec ESP [RFC4303], GRE [RFC2890] and RTP [RFC3550]. An analysis of the sequence number field can yield the lost number of packets. In certain cases, there might be an element of discovery and synchronization of the flow itself before the measurement can be made. An example of this can be found for RTP flows running on ephemeral UDP port numbers. In these cases,

reporting 0 as packet loss would be misleading and the value 0xFFFFFFFF MUST be used in cases where the packet loss value cannot be determined. In the case of a monitor interval where synchronization was achieved mid-interval, the loss packet counter MAY be used to represent the remainder of the interval. As this metric is a deltaCounter, the number of loss packets only represent the observation within the reporting interval. Due to the dependency on the arrival of a packet with a sequence number to calculate loss, the loss calculation may be indefinitely delayed if no more packets arrive at all. For the case of RTP, in addition to the 16 bit sequence number field in RFC3550, there is also the additional 16-bit high-order sequence number field (for a total of 32-bit seq number space) that is used in RFC3497 [RFC3497]. RFC3497 traffic runs at a very high rate and the 32-bit field allow for additional time for wrapping (21 seconds). So, a loss span of greater than 21 seconds measured only by the 16-bit field will lead to inaccurate reporting. In the case of secure RTP [RFC3711], the relevant portion of the RTP header is in the clear and lost packet counting can still be performed. It is important to note that the sequence number space is unique per RTP SSRC. Therefore it is important to track the high sequence number seen on a per SSRC-5-tuple basis. There may be multiple SSRCs in a single 5-tuple. Certain applications inject non-RTP traffic into the same 5-tuple as the media stream. RTCP packets may be seen in the same 5-tuple as the RTP stream [RFC5761], and STUN [RFC5389] packets may also be seen. The loss detection should ignore these packets. There may be spans within the network where header compression schemes such as [RFC2508] are used. In cases where the measurement device is terminating the compression, and the measurement implementation does not support calculation of the metric the value 0xFFFFFFFF MUST be reported. In other cases the measurement point may be at a midpoint of the header compression network span. Depending on the mechanics of header compression, sequencing information may be present and it is possible to calculate the metric. In such cases the implementation SHOULD perform the calculation and report the metric.

Units of Measurement: packets

Measurment Timing To be able to calculate this metric a continuous set of the flow’s packets (as each would have an incrementing sequence number) needs to be monitored. Therefore, per-packet sampling would prevent this metric from being calculated. However, there are other sampling methodologies that might be usable. It is possible to generate sampled metrics by sampling spans of continuous packets, however a portion of the span may have to be utilized for resynchronization of the sequence number. Another form of acceptable sampling would be at the flow level.

4.1.2. perfPacketExpected

Name: perfPacketExpected

Description: The number of packets there were expected within a monitoring interval.

Observation Point: The observation can be made anywhere along the media path or on the endpoints them selves. The observation is only relevant in a unidirectional sense.

Element Data Type: unsigned32

Element Semantics: deltaCounter

Element Units: none

Element Range Begin: 0

Element Range End: 0xFFFFFFFE

Element Id: TBDperfPacketExpected

Status: current

Use and Applications The perfPacketExpected is a mid-calculation metric used in the calculation of perfPacketLossRate.

Calculation Method: The subtraction of the last sequence number from the first sequence number in monitoring interval yields the expected count. As discussed with perfPacketLost, there might be a delay due to synchronization with the flow's sequence numbers and in such times the value of the metric should be set to 0xFFFFFFFE. Care has to be taken to account for cases where the packet's sequence number field wraps. For RTP, the expected count calculation formula can be found in Appendix A.3 of [RFC3550]. Refer to the perfPacketLoss metric regarding considerations for header compression. The value 0xFFFF is used to represent cases where the metric could not be calculated.

Units of Measurement: packets

Measurment Timing Same considerations as perfPacketLoss

4.1.3. perfPacketLossRate

Name: perfPacketLossRate

Description: Percentage of number of packets lost out of the total set of packets sent.

Observation Point: The observation can be made anywhere along the media path or on the endpoints them selves. The observation is only relevant in a unidirectional sense.

Element Data Type: unsigned16

Element Semantics: quantity

Element Units: none

Element Range Begin: 0

Element Range End: 0xFFFE

Element Id: TBDperfPacketLossRate

Status: current

Use and Applications The perfPacketLossRate metric can be used to normalize the perfPacketLoss metric to handle cases where different flows are running at different packet per second (PPS) rates. Due to the normalization, comparisons can now be made against thresholds (for creating alerts, etc.). In addition, the percentage form of the metric allows for comparisons against other flows at the same observation point to determine if there is an equal bias for drops between the flows. Otherwise, the perfoPacketLossRate is used in same way as perfPacketLoss.

Calculation Method: The number of lost packets divided by the number of expected packets in an interval period multiplied by 100. In cases where perfPacketLoss is unknown (for example due to synchronization issues), the perfPacketLossRate would also be unknown. In such cases perfPacketLossRate MUST be set to 0xFFFF. If there are multiple flows whose loss rate is being aggregated, then the average of the individual flows is used. Refer to the perfPacketLoss metric regarding considerations for header compression. The value 0xFFFF is used to represent cases where the metric could not be calculated.

Units of Measurement: percentage

Measurment Timing Same notes as perfPacketLossRate

4.1.4. perfPacketLossEvent

Name: perfPacketLossEvent

Description: The packet loss event metric reports the number of continuous sets of packets that were lost in the reporting interval.

Observation Point: The observation can be made anywhere along the media path or on the endpoints them selves. The observation is only relevant in a unidirectional sense.

Element Data Type: unsigned32

Element Semantics: deltaCounter

Element Units: packets

Element Range Begin: 0

Element Range End: 0xFFFFFFFE

Element Id: TBDperfPacketLossEvent

Status: current

Use and Applications The perfPacketLossEvent metric can provide loss information for protocols that do not implement per packet sequencing. Similarly to the perfPacketLoss metric, the packet loss event metric can be used to determine if there is a network impairment that is causing packet loss upstream of the measurement point. In cases where both the perfPacketLoss and perfPacketLossEvent metric are available, the ratio between the packet loss and packet event count can provide the average loss length. The average loss length provides additional information regarding the cause of the loss. For example, a dirty fiber connection might have a low average loss length, while a routing protocol convergence will have a high loss length.

Calculation Method: This data value is a simplified version of the Lost Packets metric. Whereas Lost Packets counts individual packet loss, the 'loss event count' metric counts sets of packets that are lost. For example, in the case of a sequence of packets: 1,3,6,7,10 the packets marked 2,4,5,8 and 9 are lost. So, a total

of 5 packets are lost. This same sequence translates to 3 loss events: (2), (4,5) and (8,9). In the case of RTP, the sequence number in the RTP header can be used to identify loss events. Certain protocols such as TCP and UDP+MPEG2-TS encapsulation in IP have sequencing information, but the sequence field is incremented by individual IP packets. As a side note, in the case of UDP+ MPEG2-TS encapsulation the simple use of RTP+MPEG2-TS via [RFC2250] results in the avaliability of the more granular perfPacketLoss metrics. In these cases, the perfPacketLoss metric cannot be calculated but the perfPacketLossEvent can be calculated and can provide detection of loss. The value 0xFFFFFFFF is used to represent non-applicable cases such as lack of sequence number synchronization. Many of the same considerations as for perfPacketLoss apply to perfPacketLoss event. Please refer to the Calculation Method section of the perfPacketLoss.

Units of Measurement: event counts

Measurment Timing Please refer to the measurement timing section of perfPacketLoss.

4.1.5. perfPacketInterArrivalJitterAvg

Name: perfPacketInterArrivalJitterAvg

Description: This metric measures the absolute deviation of the difference in packet spacing at the measurement point compared to the packet spacing at the sender.

Observation Point: The observation can be made anywhere along the media path or on the receiver. The observation is only relevant in a unidirectional sense.

Element Data Type: unsigned32

Element Semantics: quantity

Element Units: microseconds

Element Range Begin: 0

Element Range End: 0xFFFFFFFE

Element Id: TBDperfPacketInterArrivalJitterAvg

Status: current

Use and Applications The inter arrival jitter data value can be used be network operator to determine the network's impact to the spacing in between a media stream's packets as they traverse the network. For example, in the case of media applications, the receiving end system is expecting these packets to come in at a particular periodicity and large deviations may result in de-jitter buffers adding excessive delay, or the media packets being discarded. When the data is reported from multiple intermediate nodes, the area of the network that is having a detrimental contribution can be identified. On a non-media application level, the inter arrival jitter metrics can be used for early indication queuing contention within the network (which could lead to packet loss).

Calculation Method: The inter arrival jitter value makes use of the association of sending time with an IP packets and comparison of the arrival time on the monitoring point. In certain protocols, a representation of sending time is encoded into the header itself. For example, in the case of RTP packets, the RTP header's timestamps field represents encoder clock ticks-- which are representations of time. Similarly, in the case of TCP options encode absolute timestamps values. For RTP the calculation method can be found in Appendix A of [RFC3550]. It should be noted that the RFC3550 calculation is on the last 16 packets measured. The most recent value calculated SHOULD be reported at the end of the monitoring interval. The range of the jitter values during the monitoring interval can be reported using perfPacketInterArrivalJitterMin and perfPacketInterArrivalJitterMax. Similarly to the perfPacketLoss case there may be periods of time where the jitter value cannot be calculated. In these cases, the 0xFFFFFFFF value should be used to convey the lack of availability of the metric. As mentioned earlier, the RTP header timestamps is actually a 'sample-stamp' (ie clicks) from the encoder's clock. The frequency of the clock is dependent on the codec. Some codecs (eg AAC-LD) support multiple possible frequencies one of which is then selected for the media-stream. The mapping to clock rate can be performed via mapping from the static RTP payload type (RTP-PT), but newer codecs are make use of the dynamic payload type range and the RTP-PT (in the dynamic case) cannot be used to determine the clock frequency. There are various methods by which the clock frequency (deep packet inspection of the signalling, manual configuration, etc.) can be associated to the calculation method. The frequency should be locked in the metering layer to a unique combination of the IP source, IP destination, IP protocol layer-4 ports, RTP-PT and SSRC. By strict RFC3550 definition, the SSRC is set to a

specific encoder clock and it is the SSRC that should be tracked rather than payload type. However, in recent discussions it has been noted that there are RTP implementations that might change the encoder clock frequency while maintaining the SSRC value. An encoder frequency change will be accompanied by a different RTP-PT.

Units of Measurement: microseconds

Measurment Timing Please refer to the measurement timing section of perfPacketLoss.

4.1.6. perfPacketInterArrivalJitterMin

Name: perfPacketInterArrivalJitterMin

Description: This metric measures the minimum value the calculation used for perfPacketInterArrivalJitterAvg within the monitoring interval.

Observation Point: The observation can be made anywhere along the media path or on the receiver. The observation is only relevant in a unidirectional sense.

Element Data Type: unsigned32

Element Semantics: quantity

Element Units: microseconds

Element Range Begin: 0

Element Range End: 0xFFFFFFFE

Element Id: TBDperfPacketInterArrivalJitterMin

Status: current

Use and Applications Please refer to the 'Use and Applications' section of perfPacketInterArrivalJitterAvg. This specific metric, along with perfPacketInterArrivalJitterMax, is to capture the range of measurements observed within a monitoring interval as the average function may hide extremes.

Calculation Method: Please see the perfPacketInterArrivalJitterAvg section for general calculation section. The average calculation is evaluated on a running basis over the last 16 packets and the entire monitoring interval is not covered. In this metric, the

minimum value is taken over the entire monitoring interval.

Units of Measurement: microseconds

Measurment Timing Please refer to the measurement timing section of perfPacketLoss.

4.1.7. perfPacketInterArrivalJitterMax

Name: perfPacketInterArrivalJitterMax

Description: This metric measures the maximum value the calculation used for perfPacketInterArrivalJitterAvg within the monitoring interval.

Observation Point: The observation can be made anywhere along the media path or on the receiver. The observation is only relevant in a unidirectional sense.

Element Data Type: unsigned32

Element Semantics: quantity

Element Units: microseconds

Element Range Begin: 0

Element Range End: 0xFFFFFFFE

Element Id: TBDperfPacketInterArrivalJitterMax

Status: current

Use and Applications Please refer to the 'Use and Applications' section of perfPacketInterArrivalJitterAvg. This specific metric, along with perfPacketInterArrivalJitterMin, is to capture the range of measurements observed within a monitoring interval as the average function may hide extremes.

Calculation Method: Please see the perfPacketInterArrivalJitterAvg section for general calculation section. The average calculation is evaluated on a running basis over the last 16 packets and the entire monitoring interval is not covered. In this metric, the maximum value is taken over the entire monitoring interval.

Units of Measurement: microseconds

Measurment Timing Please refer to the measurement timing section of perfPacketLoss.

4.2. User and Application Layer

4.2.1. perfSessionSetupDelay

Name: perfSessionSetupDelay

Description: The Session Setup Delay metric reports the time taken from a request being initiated by a host/endpoint to the response (or request indicator) to the request being observed. This metric is defined in [RFC4710], however the units have been updated to microseconds.

Observation Point: This metric needs to be calculated where both request and response can be observed. This could be at network choke points, application proxies, or within the end systems themselves.

Element Data Type: unsigned32

Element Semantics: quantity

Element Units: microseconds

Element Range Begin: 0

Element Range End: 0xFFFFFFFE

Element Id: TBDperfSessionSetupDelay

Status: current

Use and Applications The session setup delay metric can measure the end user initial wait experience as seen from the network transaction level. The value will not only include the network flight time, but also includes the server response time and may be used to alert the operator in cases where the overall service is overloaded and thus sluggish, or within normal operating values.

Calculation Method: Measure distance in time between the first bit of request and the first bit of the response. For the case of SIP, please see Section 4.3.1 of [I-D.ietf-pmol-sip-perf-metrics]

Units of Measurement: microseconds

Measurment Timing This measurement can be sampled on a session by session basis. It may be advisable to set sample targets on a per source range - to destination basis. Due to the nature of measurement intervals, there may be a period of time (and thus measurement reports) in which the perfSessionSetupDelay value has not been calculated. In these cases the value 0xFFFFFFFE MUST be used and can be interpreted to mean not applicable. For measurement intervals after perfSessionSetupDelay has been calculated and the existing calculated perfSessionSetupDelay value SHOULD be sent if reporting only on that single session. However, if multiple sessions are summarized in the report then the average for perfSessionSetupDelay values calculated in the most recent interval SHOULD be used. The intention with this behavior is to acknowledge that the value has not bee calculated, and when it has provide the freshest values available.

4.3. Contextual Elements

4.3.1. mediaRTPSSRC

Name: mediaRTPSSRC

Description: Value of the synchronization source (SSRC) field in the RTP header of the flow. This field is defined in [RFC3550]

Observation Point: This metric can be gleaned from the RTP packets directly, so the observation point needs to on the flow path or within the endpoints.

Element Data Type: unsigned32

Element Semantics: identifier

Element Units: octets

Element Range Begin: 0

Element Range End: 0xFFFFFFFE

Element Id: TBDmediaRTPSSRC

Status: current

Use and Applications The RTP SSRC value denotes a specific media stream. As such when trying to differentiate media stream problems between session participants the SSRC field is needed.

Calculation Method: Copy from RTP header's SSRC field as defined in [RFC3550]. In the case of a non-RTP flow, or the time period in which the flow has not been verified to be a RTP flow the value 0xFFFFFFFE MUST be reported.

Units of Measurement: identifier

Measurment Timing It is possible that the SSRC may have be renegotiated mid-session due to collisions with other RTP senders.

4.3.2. mediaRTPPayloadType

Name: mediaRTPPayloadType

Description: The value of the RTP Payload Type Field as seen in the RTP header of the flow. This field is defined in [RFC3550]

Observation Point: This metric can be gleaned from the RTP packets directly, so the observation point needs to on the flow path or within the endpoints.

Element Data Type: unsigned16

Element Semantics: identifier

Element Units: octets

Element Range Begin: 0

Element Range End: 0xFF

Element Id: TBDmediaRTPPayloadType

Status: current

Use and Applications The RTP PT conveys the payload format and media encoding used in the RTP payload. For simple cases, where the RTP PT is from the statically defined range this can lead to an understanding of type of media codec used. With the knowledge of the codec being used the degree of media impairment (given loss values and jitter) can be estimated better. However, for more recent codecs, the RTP dynamic range is used. In these cases the RTP payload values are dynamically negotiated. In the case of a non-RTP flow, or the time period in which the flow has not been

verified to be a RTP flow, the value 0xFFFF MUST be reported.

Calculation Method: Copy from RTP header's RTP-PT field as defined in [RFC3550]

Units of Measurement: identifier

Measurment Timing

4.3.3. mediaCodec

Name: mediaCodec

Description: The media codec used in the flow.

Observation Point: The ideal location of this metric is on the media generators and consumers. However, given application inspection or static configuration it is possible that intermediate nodes are able to generate codec information.

Element Data Type: string

Element Semantics: identifier

Element Units: octets

Element Id: TBDmediaCodec

Status: current

Use and Applications The media codec value conveys the name of the codec used to encode the media in the flow being monitored. Simply reporting loss and jitter measurements are useful for detection of network problems. However, judging the degree of the impact on the audio/video experience needs additional information. The most basic information is the codec being used which when coupled with per-codec knowledge of sensitivity to the transport metrics a better idea of the experience can be gained.

Calculation Method: The valid values for the mediaCodec are listed on the IANA media-types registry. Analysis of the RTP payload type may lead to the determination of the media codec. However, with the use of the RTP dynamic payload type range the media information is not encoded into the data packet. For these cases, intermediate nodes may need to perform inspection of the signalling (SIP, H.323, RTSP, etc.). In cases where the mediaCodec cannot be determined, the value 'unknown' MUST be used.

Units of Measurement: identifier

Measurment Timing

5. Security Considerations

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

6. IANA Considerations

This document requires an elements assignment to be made by IANA.

7. References

7.1. Normative References

[RFC5101] Claise, B., "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information", RFC 5101, January 2008.

[RFC5610] Boschi, E., Trammell, B., Mark, L., and T. Zseby, "Exporting Type Information for IP Flow Information Export (IPFIX) Information Elements", RFC 5610, July 2009.

[RFC4710] Siddiqui, A., Romascanu, D., and E. Golovinsky, "Real-time Application Quality-of-Service Monitoring (RAQMON) Framework", RFC 4710, October 2006.

[RFC5102] Quittek, J., Bryant, S., Claise, B., Aitken, P., and J. Meyer, "Information Model for IP Flow Information Export", RFC 5102, January 2008.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.

[RFC3497] Gharai, L., Perkins, C., Goncher, G., and A. Mankin, "RTP Payload Format for Society of Motion Picture and Television Engineers (SMPTE) 292M Video", RFC 3497, March 2003.

[RFC5389] Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, "Session Traversal Utilities for NAT (STUN)", RFC 5389, October 2008.

[I-D.ietf-pmol-sip-perf-metrics]
Malas, D. and A. Morton, "Basic Telephony SIP End-to-End Performance Metrics", draft-ietf-pmol-sip-perf-metrics-07 (work in progress), September 2010.

[iana-ipfix-assignments]
Internet Assigned Numbers Authority, "IP Flow Information Export Information Elements (http://www.iana.org/assignments/ipfix/ipfix.xml)".

7.2. Informative References

[I-D.ietf-pmol-metrics-framework]
Clark, A. and B. Claise, "Guidelines for Considering New Performance Metric Development", draft-ietf-pmol-metrics-framework-05 (work in progress), October 2010.

[I-D.trammell-ipfix-ie-doctors]
Trammell, B. and B. Claise, "Guidelines for Authors and Reviewers of IPFIX Information Elements", draft-trammell-ipfix-ie-doctors-00 (work in progress), October 2010.

[RFC2508] Casner, S. and V. Jacobson, "Compressing IP/UDP/RTP Headers for Low-Speed Serial Links", RFC 2508, February 1999.

[RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, March 2004.

[RFC2250] Hoffman, D., Fernando, G., Goyal, V., and M. Civanlar, "RTP Payload Format for MPEG1/MPEG2 Video", RFC 2250, January 1998.

[RFC2890] Dommety, G., "Key and Sequence Number Extensions to GRE", RFC 2890, September 2000.

[RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, December 2005.

[RFC5761] Perkins, C. and M. Westerlund, "Multiplexing RTP Data and Control Packets on a Single Port", RFC 5761, April 2010.

[I-D.huici-ipfix-sipfix]
Huici, F., Niccolini, S., and S. Anderson, "SIPFIX: Use Cases and Problem Statement for VoIP Monitoring and

Exporting", draft-huici-ipfix-sipfix-00 (work in progress), June 2009.

[nProbe] "probe - NetFlow/IPFIX Network Probe (http://www.ntop.org/nProbe.html)".

[RFC2321] Bressen, A., "RITA -- The Reliable Internetwork Troubleshooting Agent", RFC 2321, April 1998.

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

[RFC5477] Dietz, T., Claise, B., Aitken, P., Dressler, F., and G. Carle, "Information Model for Packet Sampling Exports", RFC 5477, March 2009.

[VoIP-monitor] L. Chang-Yong, H. Kim, K. Ko, J. Jim, and H. Jeong, "A VoIP Traffic Monitoring System based on NetFlow v9, International Journal of Advanced Science and Technology, vol. 4, Mar. 2009".

Author's Address

Aamer Akhter
Cisco Systems, Inc.
7025 Kit Creek Road
RTP, NC 27709
USA

Email: aakhter@cisco.com

IPFIX Working Group
Internet-Draft
Intended Status: Informational
Expires: September 9, 2011

B. Claise
P. Aitken
N. Ben-Dvora
Cisco Systems, Inc.
March 9, 2011

Export of Application Information in IPFIX
draft-claise-export-application-info-in-ipfix-01

Status of this Memo

Abstract

This document specifies an extension to the IPFIX information model specified in [RFC5102] to export application information.

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

List of Figures and Tables

1. Overview

1.1. IPFIX Documents Overview

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

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

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

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

In order to gain a level of confidence in the IPFIX implementation, probe the conformity and robustness, and allow interoperability, the Guidelines for IPFIX Testing [RFC5471] presents a list of tests for implementers of compliant Exporting Processes and Collecting Processes.

The Bidirectional Flow Export [RFC5103] specifies a method for exporting bidirectional flow (biflow) information using the IP Flow Information Export (IPFIX) protocol, representing each Biflow using a single Flow Record.

The "Reducing Redundancy in IP Flow Information Export (IPFIX) and Packet Sampling (PSAMP) Reports" [RFC5473] specifies a bandwidth saving method for exporting Flow or packet information, by separating information common to several Flow Records from information specific to an individual Flow Record: common Flow information is exported only once.

2. Introduction

Today service providers and network administrators are looking for visibility into the packet content rather than just the packet header. Some network devices Metering Processes inspect the packet content and identify the applications that are utilizing the network traffic. Applications in this context are defined as networking protocols used by networking processes that exchange packets between them (such as the web applications, peer to peer applications, file transfer, e-mail applications, etc.). Combined with other information elements, some of which being application specific, the applications can be further characterized. Examples include: web application to a specific domain, per user specific traffic, a video application with a specific codec, etc...

The application identification is based on different kind of methods or even a combination of such methods:

1. L2 protocols (such as ARP, PPP, LLDP)
2. IP protocols (such as ICMP, IGMP, GRE)
3. TCP or UDP ports (such as HTTP, Telnet, FTP)
4. Application layer header (of the application to be identified)
5. Packet data content
6. Packets and traffic behavior

The exact application identification methods are part of the Metering Process internals that aims to provide an accurate identification with a minimum false identification. This task requires a sophisticated Metering Process since the protocols do not behave in a standard manner.

1. Applications use port obfuscation where the application run on different port than the IANA assigned one. For example a HTTP server might run a TCP port 23 (assigned to telnet in [IANA-PORTS])
2. IANA does not accurately reflect how certain ports are "commonly" used today. Some ports are reserved, but the

application either never became prevalent or is not in use today.

3. The application behavior and identification logic become more and more complex

For that reason, such Metering Processes usually detect application based on multiple mechanisms in parallel. Detecting applications based only on port matching might wrongly identify the traffic. Note that this example stresses the need for the engine strength. If the Metering Process is capable of detecting applications more accurately it is considered as stronger and more accurate.

Similarly, a reporting mechanism that uses L4 port based applications only, such as L4:<known port>, would have a similar issues. The reporting system should be capable of reporting the applications classified using all types for mechanisms. In particular applications that does not have any IANA port definition. While a mechanism to export application information should be defined, the L4 port being in use must be exported using the destination port (destinationTransportPort at [IANA-IPFIX]) in the corresponding NetFlow record.

Cisco Systems uses the IPFIX application tag as described in section 4. to export the application information with the IPFIX protocol [RFC5101].

Application could be defined at different OSI layers, from the layer 2 to the layer 7. Examples: Cisco Discovery Protocol is layer 2 application, ICMP is layer 3 application [IANA-PROTO], HTTP is layer 4 application [IANA-PORTS], and skype is layer 7.

While an ideal solution would be an IANA registry for applications above (or inside the payload of) the well known ports [IANA-PORTS], this solution is not always possible as the some applications require well known specifications. Therefore, some reverse engineering is required, as well as a ubiquitous language for application identification. Clearly not realistic.

As this specification focuses on the application information encoding, this document doesn’t contain an application registry for non IANA applications. However, a reference to the Cisco assigned numbers for the Application Tag and the different attribute assignments can be found at [CISCO].

2.1. Application Information Use Cases

There are several use cases on which the application information is used:

1. Network Visibility

   This is one of the main use cases for using the application information. This use case is also called application visibility. Network administrators are using such application visibility to understand the main network consumers, network trends and user behavior.

2. Billing Services

   In some cases, network providers are willing to bill different applications differently. For example, provide different billing for VoIP and Web browsing.

3. Congestion Control

   While the traffic demand is increasing (mainly due to the high usage of peer to peer applications, video applications and web download applications), the providers revenue doesn't grow. Providers are looking at a more efficient way to control and prioritize the network utilization. An application aware bandwidth control system is used to prioritize the traffic based on the applications, giving the critical applications priority over the non-critical applications.

4. Security Functions

   Application knowledge is sometimes used in security functions in order to provide comprehensive functions such as Application based firewall, URL filtering, Parental control, Intrusion detection, etc.

All of the above use cases require exporting of application information to provide the network function itself or to log the network function operation.

3. Terminology

IPFIX-specific terminology used in this document is defined in Section 2 of the IPFIX protocol specification [RFC5101]. As in

[RFC5101], these IPFIX-specific terms have the first letter of a word capitalized when used in this document.

3.1. New Terminology

Application Tag

A unique identifier for an application.

4. applicationTag Information Element Specification

This document specifies the applicationTag Information Element, which is composed of two parts:

1. 8 bits of Classification Engine ID. The Classification Engine can be considered as a specific registry for application assignment.
2. m bits of Selector ID. The Selector ID length varies depending on the engine.

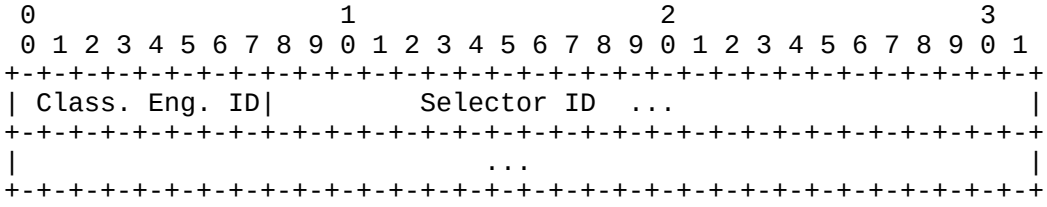


Figure 1: applicationTag Information Element

Classification Engine ID

A unique identifier for the engine which determined the Selector ID. Thus the Classification Engine ID defines the context for the Selector ID.

Selector ID

A unique identifier of the application for a specific Classification Engine ID.

Note that the Selector ID term is in sync with the PSAMP terminology. See [RFC5476], Packet Sampling (PSAMP) Protocol Specifications.

When an application is detected, the most granular application is encoded in the Application Tag: for example, ICMP would be encoded as layer 3 value 1, SNMP as layer 4 value 161, bittorent as layer 7 value 69.

The overall length of the applicationTag Information Element may be specified either in the IPFIX Template Record or by using an IPFIX Variable-Length Information Element. The receiver / decoder must respect this length rather than using the Classification Engine ID to make an assumption about the Selector ID size.

When exporting applicationTag information in IPFIX, the applicationTag SHOULD be encoded in a variable-length Information Element [RFC5101]. However, if a legacy protocol such as NetFlow version 9 is used, and this protocol doesn't support variable length Information Elements, then either multiple templates (one per applicationTag length), or a single template corresponding to the maximum sized applicationTag MUST be used. This avoids the need for multiple Template Records with different applicationTag lengths when the IPFIX variable length encoding [RFC5101] is not available.

As a consequence, although some Application Tags can be encoded in a smaller number of bytes (eg, an IANA L3 protocol encoding would take 2 bytes, while an IANA L4 port encoding would take 3 bytes), nothing prevents an Exporting Process from exporting all Application Tags with a larger fixed length.

Note that the Selector ID value is always encoded in the least significant bits as shown:

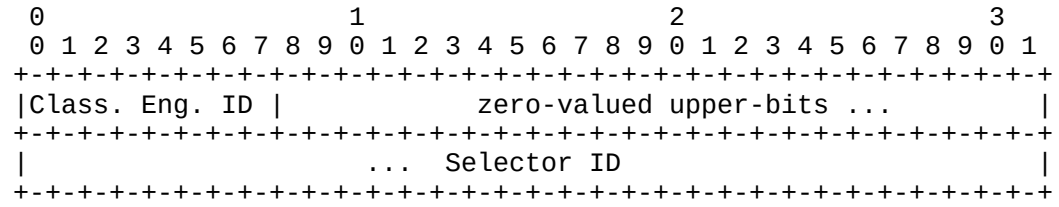


Figure 2: Selector ID encoding

4.1. Existing Classification Engine IDs

The following Engine IDs have been allocated by Cisco Systems.

Name	Value	Description
	0	Invalid.
IANA-L3	1	The IANA protocol (layer 3) number is exported in the Selector ID. See http://www.iana.org/assignments/protocol-numbers.
CANA-L3	2	Cisco Systems proprietary layer 3 definition. Cisco Systems can still export its own layer 3 protocol numbers, while waiting for IANA to assign it. The Selector ID has a global significance for all Cisco Systems devices under CANA governance. Hopefully the same IDs will be maintained after the IANA standardization.
IANA-L4	3	IANA layer 4 well-known port number is exported in the Selector ID. See http://www.iana.org/assignments/port-numbers. Note: as a flow is unidirectional, it contains the destination port in a flow from the client to the server.
CANA-L4	4	Cisco Systems proprietary layer 4 definition. Cisco Systems can still export its own layer 4 port numbers, while waiting for IANA to assign it. The Selector ID has global significance for all Cisco Systems devices under CANA governance. Hopefully the same ID will be maintained after the IANA standardization. Example: IPFIX had the port 4739 pre-assigned in the IETF draft for years. While waiting for the IANA registration, we could use this Selector ID.
	5	Reserved.
USER-Defined	6	The Selector ID represents applications defined by the user (using CLI or GUI) based on the methods described in section 2.
	7	Reserved.

	8	Reserved.
	9	Reserved.
	10	Reserved.
	11	Reserved.
CANA-L2	12	The Selector ID represents the Cisco Systems unique global layer 2 applications. The Selector ID has a global significance.
CANA-L7	13	The Selector ID represents the Cisco Systems unique global ID for the layer 7 applications. The Selector ID has global significance for all Cisco Systems devices.
	14	Reserved.
	15	Reserved.
	16	Reserved.
	17 to 254	Available.
MAX	255	255 is the maximum Engine ID.

Table 1: Existing Classification Engine IDs

Note 1: "CANA = Cisco Systems Assigned Number Authority", Cisco Systems's version of IANA for internal IDs.

Note 2: This is an extensible list, and new Classification Engine IDs may be allocated at any time. See [CISCO] for the latest version.

4.2. Options Template Record for the Application Name

For engines which specify locally unique Application Tags (which means unique per engine and per router), an Options Template Record (see [RFC5101]) MUST be used to export the correspondence between the Application Tag, the Application Name, and the Application Description. This is called the "options application-table". For engines which specify globally unique Application Tags, an Options Template Record SHOULD be used to export the correspondence between the Application Tag, the Application Name and the Application Description, unless the mapping is hardcoded in the NetFlow Collector, or known out of band (for example, by polling a MIB).

4.3. Resolving IANA L4 port collisions

Even if the IANA L4 ports usually point to the same protocols for both UDP, TCP or other transport types, there are some exceptions. The following table lists 10 ports that have different protocols assigned for TCP and UDP:

```
exec            512/tcp    remote process execution;
#                          authentication performed using
#                          passwords and UNIX login names
comsat/biff     512/udp    used by mail system to notify users
#                          of new mail received; currently
#                          receives messages only from
#                          processes on the same machine
login           513/tcp    remote login a la telnet;
#                          automatic authentication performed
#                          based on priviledged port numbers
#                          and distributed data bases which
#                          identify "authentication domains"
who             513/udp    maintains data bases showing who's
#                          logged in to machines on a local
#                          net and the load average of the
#                          machine
shell           514/tcp    cmd
#                          like exec, but automatic
authentication
#                          is performed as for login server
syslog          514/udp
oob-ws-https    664/tcp    DMTF out-of-band secure web services
#                          management protocol
#                          Jim Davis
<jim.davis&wbemsolutions.com>
#                          June 2007
asf-secure-rmcp 664/udp    ASF Secure Remote Management
#                          and Control Protocol
rfile           750/tcp
kerberos-iv     750/udp    kerberos version iv
submit          773/tcp
notify          773/udp
rpasswd         774/tcp
acmaint_dbd     774/udp
entomb          775/tcp
acmaint_transd  775/udp
busboy          998/tcp
puparp          998/udp
garcon          999/tcp
```

```
  applix          999/udp    Applix ac
```

Table 2: IANA layer 4 port collisions between UDP and TCP

The following table lists 19 ports that have different protocols assigned for TCP and SCTP:

```
#               3097/tcp    Reserved
itu-bicc-stc    3097/sctp   ITU-T Q.1902.1/Q.2150.3
                            Greg Sidebottom<gregside&home.com>
#               5090/tcp    <not assigned>
car             5090/sctp   Candidate AR
#               5091/tcp    <not assigned>
cxtp            5091/sctp   Context Transfer Protocol
                           RFC 4065 - July 2005
#               6704/tcp    Reserved
frc-hp          6704/sctp   ForCES HP (High Priority) channel
#                           [RFC5811]
#               6705/tcp    Reserved
frc-mp          6705/sctp   ForCES MP (Medium Priority) channel
#                           [RFC5811]
#               6706/tcp    Reserved
frc-lp          6706/sctp   ForCES LP (Low priority) channel
#                          [RFC5811]
#               9082/tcp    <not assigned>
lcs-ap          9082/sctp   LCS Application Protocol
#                           Kimmo Kymalainen
                            <kimmo.kymalainen&etsi.org>
                            04 June 2010
#               9902/tcp    <not assigned>
enrp-sctp-tls   9902/sctp   enrp/tls server channel
#                          [RFC5353]
#               11997/tcp   <not assigned>
#               11998/tcp   <not assigned>
#               11999/tcp   <not assigned>
wmereceiving    11997/sctp  WorldMailExpress
wmedistribution 11998/sctp  WorldMailExpress
wmereporting    11999/sctp  WorldMailExpress
                           Greg Foutz<gregf&adminovation.com>
                            March 2006
#               25471/tcp   <not assigned>
rna             25471/sctp  RNSAP User Adaptation for Iurh
```

#		Dario S. Tonesi <dario.tonesi&nsn.com> 07 February 2011
#	29118/tcp	Reserved
sgsap	29118/sctp	SGsAP in 3GPP
#	29168/tcp	Reserved
sbcap	29168/sctp	SBcAP in 3GPP
#	29169/tcp	<not assigned>
iuhsctpassoc	29169/sctp	HNBAP and RUA Common Association John Meredith<John.Meredith&etsi.org> 08 September 2009
#	36412/tcp	<not assigned>
s1-control	36412/sctp	S1-Control Plane (3GPP)
#		KimmoKymalainen <kimmo.kymalainen&etsi.org> 01 September 2009
#	36422/tcp	<not assigned>
x2-control	36422/sctp	X2-Control Plane (3GPP)
#		Kimmo Kymalainen <kimmo.kymalainen&etsi.org> 01 September 2009
#	36443/tcp	<not assigned>
m2ap	36443/sctp	M2 Application Part
#		Dario S. Tonesi <dario.tonesi&nsn.com> 07 February 2011
#	36444/tcp	<not assigned>
m3ap	36444/sctp	M3 Application Part
#		Dario S. Tonesi <dario.tonesi&nsn.com> 07 February 2011

Table 3: IANA layer 4 port collisions between SCTP and TCP

Instead of imposing the transport protocol (UDP/TCP/SCTP/etc.) in the scope of the "options application-table" Options Template for all applications (on top of having the transport protocol as key-field in the Flow Record definition), we define that the L4 application is always TCP related, by convention. So, whenever the Collector has a conflict in looking up IANA, it would choose the TCP choice. As a result, the UDP L4 applications from Table

2 and the SCTP L4 applications from table 3 are assigned in the Cisco L7 Application Tag range (ie, under Classification Engine ID 13):

Currently, there are no discrepancies between the well known ports for TCP and DCCP.

5. Grouping the Applications with the Attributes

Due to the high number of different application tags, categorizing them into groups offers the benefits of easier reporting and action, such as QoS policies. Indeed, most applications with the same characteristics should be treated the same way; for example, all video traffic.

Attributes are statically assigned per application tag and are independent of the traffic. The attributes are listed below:

Name	Description
Category	An attribute that provides a first level categorization for each application tag. Examples include: browsing, email, file-sharing, gaming, instant messaging, voice-and-video, etc... The category attribute is encoded by the ApplicationCategoryName Information Element.
Sub-Category	An attribute that provides a second level categorization for each application tag. Examples include: backup-systems, client-server, database, routing-protocol, etc... The sub-category attribute is encoded by the ApplicationSubCategoryName Information Element.
Application-Group	An attribute that groups multiple application tags that belong to the same networking application. For example, the ftp-group contain the ftp-data (port 20), ftp (port 20), ni-ftp (port 47), sftp (port 115),

	bftp (port 152), ftp-agent(port 574), ftps-data (port 989). The application-group attribute is encoded by the ApplicationGroupName Information Element.
P2P-Technology	Specifies if the application tag is based on peer-to-peer technology. The P2P-technology attribute is encoded by the p2pTechnology Information Element.
Tunnel-Technology	Specifies if the application tag is used as a tunnel technology. The tunnel-technology attribute is encoded by the tunnelTechnolgoy Information Element.
Encrypted	Specifies if the application tag is an encrypted networking protocol. The encrypted attribute is encoded by the encryptedTechnology Information Element.

Table 4: Existing Application Tag Static Attributes

Every application is assigned to one ApplicationCategoryName, one ApplicationSubCategoryName, one ApplicationGroupName, has one p2pTechnology, one tunnelTechnolgoy, and one encryptedTechnology.

5.1. Options Template Record for the Attribute Values

An Options Template Record (see [RFC5101]) is used to export the correspondence between each Application Tag and its related Attribute values. An alternative way for the Collecting Process to learn the correspondence is to populate these mappings out of band, for example, by loading a CSV file containing the correspondence table.

The Attributes Option Template contains the ApplicationTag as a scope field, followed by the ApplicationCategoryName, the ApplicationSubCategoryName, the ApplicationGroupName, the p2pTechnology, the tunnelTechnolgoy, and the encryptedTechnology Information Elements.

A list of attributes may conveniently be exported using a subTemplateList per [IPFIX-STRUCT].

An example is given in section 6.8 below.

6. Application Tag Examples

The following examples are created solely for the purpose of illustrating how the extensions proposed in this document are encoded.

6.1. Example 1: Layer 2 Protocol

From the list of Classification Engine IDs in Table 1, we can see that the layer 2 Classification Engine ID is 12:

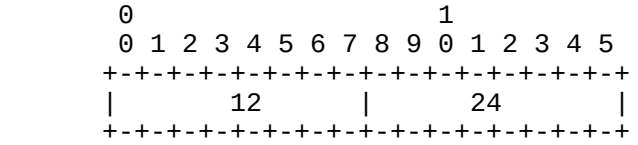
L2	12	The Selector ID represents the layer 2 applications. The Selector ID has a global significance.

From the list of layer 2 protocols at [cisco], we can see that PPP has the value 24:

NAME	Selector ID
ppp	24

So, in the case of layer 2 protocol PPP, the Classification Engine ID is 12 while the Selector ID has the value 24.

Therefore the Application Tag is encoded as:



So the Application Tag has the value of 3097. Instead of representing the Application Tag in hexadecimal format, the format ’12...24’ is used for simplicity in the examples below.

Flexible NetFlow creates a Template Record with a few Information Elements: amongst other things, the Application Tag. For example:

- sourceIPv4Address (key field)
- destinationIPv4Address (key field)
- ipDiffServCodePoint (key field)
- applicationTag (key field)
- octetTotalCount (non key field)

For example, a Flow Record corresponding to the above Template Record may contain:

```
{ sourceIPv4Address=1.1.1.1, destinationIPv4Address=2.2.2.2,
  ipDiffServCodePoint=0, applicationTag='12...24',
  octetTotalCount=123456 }
```

The Collector has all the required information to determine that the application is PPP, because the Application Tag uses a global and well know registry, ie the IANA protocol number. The 24 value is globally unique within Cisco Systems for Classification Engine ID 12, so the Collector can determine which application is represented by the Application Tag by loading the registry out of band.

6.2. Example 2: Standardized IANA Layer 3 Protocol

From the list of Classification Engine IDs in Table 1, we can see that the IANA layer 3 Classification Engine ID is 1:

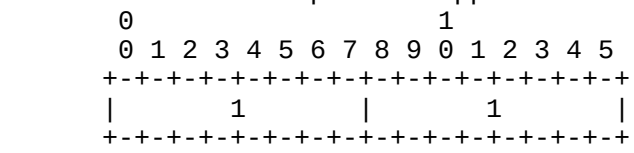
IANA-L3	1	The IANA protocol (layer 3) number is exported in the Selector ID. See http://www.iana.org/assignments/protocol-numbers..

From the list of IANA layer 3 protocols (see [IANA-PROTO]), we can see that ICMP has the value 1:

Decimal	Keyword	Protocol	Reference
1	ICMP	Internet Control Message	[RFC792]

So in the case of the standardized IANA layer 3 protocol ICMP, the Classification Engine ID is 1, and the Selector ID has the value of 1.

Therefore the Application Tag is encoded as:



So the Application Tag has the value of 257. Instead of representing the Application Tag in hexadecimal format, the format '1...1' is used for simplicity in the examples below.

Flexible NetFlow creates a Template Record with a few Information Elements: amongst other things, the Application Tag. For example:

- sourceIPv4Address (key field)
- destinationIPv4Address (key field)
- ipDiffServCodePoint (key field)
- applicationTag (key field)
- octetTotalCount (non key field)

For example, a Flow Record corresponding to the above Template Record may contain:

{ sourceIPv4Address=1.1.1.1, destinationIPv4Address=2.2.2.2,
ipDiffServCodePoint=0, applicationTag='1...1',
octetTotalCount=123456 }

The Collector has all the required information to determine that the application is ICMP, because the Application Tag uses a global and well know registry, ie the IANA L3 protocol number.

6.3. Example 3: Cisco Systems Proprietary Layer 3 Protocol

Assume that Cisco Systems has specified a new layer 3 protocol called "foo".

From the list of Classification Engine IDs in Table 1, we can see that the Cisco Systems layer 3 Classification Engine ID is 2:

CANA-L3	2	Cisco Systems proprietary layer 3 definition. Cisco Systems can still export its own layer 3 protocol numbers, while waiting for IANA to assign it. The

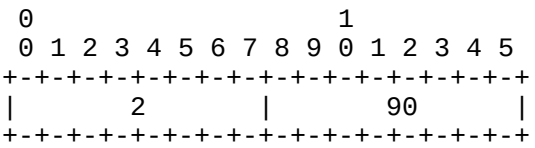
Selector ID has a global significance for all Cisco Systems devices under CANA governance. Hopefully the same IDs will be maintained after the IANA standardization.

A global registry within Cisco Systems specifies that the "foo" protocol has the value 90:

Protocol	Protocol Id
foo	90

So in the case of Cisco Systems layer 3 protocol foo, the Classification Engine ID is 2, and the Selector ID has the value of 90.

Therefore the Application Tag is encoded as:



So the Application Tag has the value of 602. Instead of representing the Application Tag in hexadecimal format, the format ’2..90’ is used for simplicity in the examples below.

Flexible NetFlow creates a Template Record with a few Information Elements: amongst other things, the Application Tag. For example:

- sourceIPv4Address (key field)
- destinationIPv4Address (key field)
- ipDiffServCodePoint (key field)
- applicationTag (key field)
- octetTotalCount (non key field)

For example, a Flow Record corresponding to the above Template Record may contain:

```
{ sourceIPv4Address=1.1.1.1, destinationIPv4Address=2.2.2.2,
  ipDiffServCodePoint=0, applicationTag=’2...90’,
  octetTotalCount=123456 }
```

Along with this Flow Record, a new Options Template Record would be exported, as shown in Section 6.7.

6.4. Example 4: Standardized IANA Layer 4 Port

From the list of Classification Engine IDs in Table 1, we can see that the IANA layer 4 Classification Engine ID is 3:

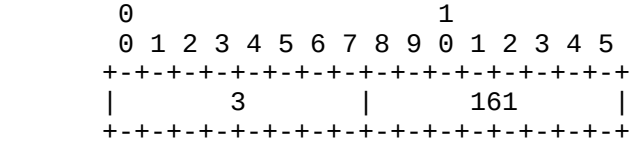
IANA-L4	3	IANA layer 4 well-known port number is exported in the selector ID. See http://www.iana.org/assignments/port-numbers. Note: as a flow is unidirectional, it contains the destination port in a flow from the client to the server.

From the list of IANA layer 4 ports (see [IANA-PORTS]), we can see that SNMP has the value 161:

Keyword	Decimal	Description
snmp	161/tcp	SNMP
snmp	161/udp	SNMP

So in the case of the standardized IANA layer 4 SNMP port, the Classification Engine ID is 3, and the Selector ID has the value of 161.

Therefore the Application Tag is encoded as:



Flexible NetFlow creates a Template Record with a few Information Elements: amongst other things, the Application Tag. For example:

- sourceIPv4Address (key field)
- destinationIPv4Address (key field)
- protocol (key field)

- ipDiffServCodePoint (key field)
- applicationTag (key field)
- octetTotalCount (non key field)

For example, a Flow Record corresponding to the above Template Record may contain:

{ sourceIPv4Address=1.1.1.1, destinationIPv4Address=2.2.2.2, protocol=17, ipDiffServCodePoint=0, applicationTag='3..161', octetTotalCount=123456 }

The Collector has all the required information to determine that the application is SNMP, because the Application Tag uses a global and well know registry, ie the IANA L4 protocol number.

6.5. Example 4: Layer 7 Application

In this example, the Metering Process has observes some Citrix traffic.

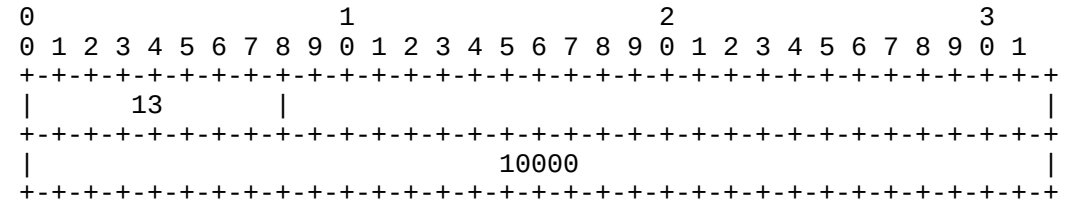
From the list of Classification Engine IDs in Table 1, we can see that the L7 unique Engine ID is 13:

L7	13	The Selector ID represents the Cisco Systems unique global ID for the layer 7 application. The Selector ID has a global significance for all Cisco Systems devices.

Suppose that the Metering Process returns the ID 10000 for Citrix traffic.

So, in the case of this Citrix application, the Classification Engine ID is 13 and the Selector ID has the value of 10000.

Therefore the Application Tag is encoded as:



So the Application Tag has the value of '13..10000'.

Note that the figure shows that the Exporting Process exports the value 10000 in 7 bytes: this is pure speculation. However, it doesn't matter as the applicationTag would be exported in a variable length Information Element.

Flexible NetFlow creates a Template Record with a few Information Elements: amongst other things, the Application Tag. For example:

- sourceIPv4Address (key field)
- destinationIPv4Address (key field)
- ipDiffServCodePoint (key field)
- applicationTag (key field)
- octetTotalCount (non key field)

For example, a Flow Record corresponding to the above Template Record may contain:

```
{ sourceIPv4Address=1.1.1.1, destinationIPv4Address=2.2.2.2,
  ipDiffServCodePoint=0, applicationTag='13...10000',
  octetTotalCount=123456 }
```

The 10000 value is globally unique within Cisco Systems, so the Collector can determine which application is represented by the Application Tag by loading the registry out of band.

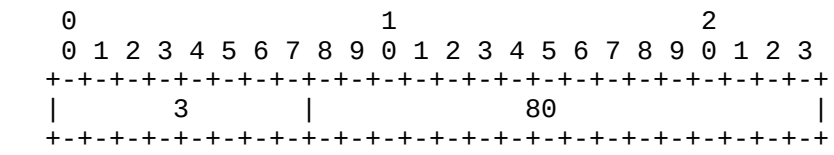
Along with this Flow Record, a new Options Template Record would be exported, as shown in Section 6.7.

6.6. Example: port Obfuscation

For example, a HTTP server might run a TCP port 23 (assigned to telnet in [IANA-PORTS]). If the Metering Process is capable of detecting HTTP in the same case, the Application Tag representation must contain HTTP. However, if the reporting application wants to determine whether or the default HTTP port 80 or 8080 was used, it must export the destination port (destinationTransportPort at [IANA-IPFIX]) in the corresponding NetFlow record.

In the case of a standardized IANA layer 4 port, the Classification Engine ID is 2, and the Selector ID has the value of 80 for HTTP (see [IANA-PORTS]).

Therefore the Application Tag is encoded as:



Flexible NetFlow creates a Template Record with a few Information Elements: amongst other things, the Application Tag. For example:

- sourceIPv4Address (key field)
- destinationIPv4Address (key field)
- protocol (key field)
- destinationTransportPort (key field)
- applicationTag (key field)
- octetTotalCount (non key field)

For example, a Flow Record corresponding to the above Template Record may contain:

```
{ sourceIPv4Address=1.1.1.1, destinationIPv4Address=2.2.2.2,
  protocol=17, destinationTransportPort=23,
  applicationTag='3..80', octetTotalCount=123456 }
```

The Collector has all the required information to determine that the application is HTTP, but runs on port 23.

6.7. Example: Application Mapping Options Template

Along with the Flow Records shown in the above examples, a new Options Template Record would be exported to express the Application Name and Application Description associated with each Application Tag.

The Options Template Record contains the following Information Elements:

1. Scope = applicationTag.

   From RFC 5101: "The scope, which is only available in the Options Template Set, gives the context of the reported Information Elements in the Data Records."

2. applicationName.

3. applicationDescription.

The Options Data Record associated with the examples above would contain, for example:

```
{ scope=applicationTag='2...90',
  applicationName="foo",
  applicationDescription="The Cisco foo protocol",

  scope=applicationTag='13...10000',
  applicationName="Citrix",
  applicationDescription="A Citrix application" }
```

When combined with the example Flow Records above, these Options Template Records tell the NetFlow collector:

1. A flow of 123456 bytes exists from sourceIPv4Address 1.1.1.1 to destinationIPv4address 2.2.2.2 with a DSCP value of 0 and an applicationTag of '12...90', which maps to the "foo" application.

2. A flow of 123456 bytes exists from sourceIPv4Address 1.1.1.1 to destinationIPv4address 2.2.2.2 with a DSCP value of 0 and an Application Tag of '13...10000', which maps to the "Citrix" application.

6.8. Example: Attributes Values Options Template Record

Along with the Flow Records shown in the above examples, a new Options Template Record is exported to express the values of the different attributes related to the Application Tags.

The Options Template Record would contain the following Information Elements:

1. Scope = applicationTag.

   From RFC 5101: "The scope, which is only available in the Options Template Set, gives the context of the reported Information Elements in the Data Records."

2. applicationCategoryName.

3. applicationSubCategoryName.

4. applicationGroupName

5. p2pTechnology

6. tunnelTechnology

7. encryptedTechnology

The Options Data Record associated with the examples above would contain, for example:

```
{ scope=applicationTag='2...90',
  applicationCategoryName="foo-category",
  applicationSubCategoryName="foo-subcategory",
  applicationGroupName="foo-group",
  p2pTechnology=NO
  tunnelTechnology=YES
  encryptedTechnology=NO
```

When combined with the example Flow Records above, these Options Template Records tell the NetFlow collector:

A flow of 123456 bytes exists from sourceIPv4Address 1.1.1.1 to destinationIPv4address 2.2.2.2 with a DSCP value of 0 and an applicationTag of '12...90', which maps to the "foo" application. This application can be characterized by the relevant attributes values.

7. IANA Considerations

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

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

EDITOR'S NOTE: the XML specification in Appendix A must be updated with the elementID values allocated below.

7.1. applicationDescription

Name: applicationDescription
Description:
  Specifies the description of an application.
Abstract Data Type: string
Data Type Semantics:
ElementId: 94
Status: current

7.2. applicationTag

Name: applicationTag
Description:
  Specifies an Application Tag.
  (EDITOR'S NOTE: reference this document).
Abstract Data Type: octetArray
Data Type Semantics: identifer
ElementId: 95
Status: current

7.3. applicationName

Name: applicationName
Description:
  Specifies the name of an application.
Abstract Data Type: string
Data Type Semantics:
ElementId: 96
Status: current

7.4. applicationCategoryName

Name: applicationCategoryName
Description:
  An attribute that provides a first level categorization for each
Application Tag.
Abstract Data Type: string
Data Type Semantics:
ElementId: <to be assigned>
Status: current

7.5. applicationSubCategoryName

Name: applicationSubCategoryName
Description:
An attribute that provides a second level categorization for each Application Tag
Abstract Data Type: string
Data Type Semantics:
ElementId: <to be assigned>
Status: current

7.6. applicationGroupName

Name: applicationGroupName
Description:
An attribute that groups multiple Application Tags that belong to the same networking application
Abstract Data Type: string
Data Type Semantics:
ElementId: <to be assigned>
Status: current

7.7. p2pTechnology

Name: p2pTechnology
Description:
Specifies if the Application Tag is based on peer-to-peer technology. Possible values are: "yes", "no", and "unassigned"
Abstract Data Type: string
Data Type Semantics:
ElementId: <to be assigned>
Status: current

7.8. tunnelTechnology

Name: tunnelTechnology
Description:
Specifies if the application tag is used as a tunnel technology. Possible values are: "yes", "no", and "unassigned"
Abstract Data Type: string
Data Type Semantics:
ElementId: <to be assigned>
Status: current

7.9. encryptedTechnology

Name: encryptedTechnology
Description:
Specifies if the application tag is an encrypted networking protocol. Possible values are: "yes", "no", and "unassigned"
Abstract Data Type: string
Data Type Semantics:
ElementId: <to be assigned>
Status: current

8. Security Considerations

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

9. References

9.1. Normative References

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

[RFC5101] Claise, B., Ed., "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information", RFC 5101, January 2008.

[RFC5102] Quittek, J., Bryant, S., Claise, B., Aitken, P., and J. Meyer, "Information Model for IP Flow Information Export", RFC 5102, January 2008.

9.2. Informative References

[RFC792] J. Postel, Internet Control Message Protocol, RFC 792, September 1981.

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

[RFC5103] Trammell, B., and E. Boschi, "Bidirectional Flow Export Using IP Flow Information Export (IPFIX)", RFC 5103, January 2008.

[RFC5470] Sadasivan, G., Brownlee, N., Claise, B., and J. Quittek, "Architecture for IP Flow Information Export", RFC 5470, March 2009.

[RFC5471] Schmoll, C., Aitken, P., and B. Claise, "Guidelines for IP Flow Information Export (IPFIX) Testing", RFC 5471, March 2009.

[RFC5473] Boschi, E., Mark, L., and B. Claise, "Reducing Redundancy in IP Flow Information Export (IPFIX) and Packet Sampling (PSAMP) Reports", RFC 5473, March 2009.

[RFC5476] Claise, B., Ed., "Packet Sampling (PSAMP) Protocol Specifications", RFC 5476, March 2009.

[IPFIX-STRUCT] Claise, B., Dhandapani, G. Aitken, P., and S. Yates, "Export of Structured Data in IPFIX", draft-ietf-ipfix-structured-data-05.txt, work in progress, March 20111

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

[IANA-PORTS] http://www.iana.org/assignments/port-numbers

[IANA-PROTO] http://www.iana.org/assignments/protocol-numbers

[CISCO] http://www.cisco.com

10. Acknowledgement

The authors would like to thank their many colleagues across Cisco Systems who made this work possible.

11. Authors' Addresses

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

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

Paul Aitken
Cisco Systems, Inc.
96 Commercial Quay
Commercial Street
Edinburgh, EH6 6LX, United Kingdom

Phone: +44 131 561 3616
EMail: paitken@cisco.com

Nir Ben-Dvora
Cisco Systems, Inc.
32 HaMelacha St.,
P.O.Box 8735, I.Z.Sapir
South Netanya, 42504
Israel

Phone: +972 9 892 7187
EMail: nirbd@cisco.com

Appendix A. Additions to XML Specification of IPFIX Information Elements

This appendix contains additions to the machine-readable description of the IPFIX information model coded in XML in Appendix A and Appendix B in [RFC5102]. Note that this appendix is of informational nature, while the text in Section 7. (generated from this appendix) is normative.

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

```
<field name="applicationDescription"
         dataType="string"
         group="application"
         elementId="94" applicability="all" status="current">
    <description>
      <paragraph>
         Specifies the description of an application.
      </paragraph>
    </description>
  </field>

  <field name="applicationTag"
         dataType="octetArray"
         group="application"
         dataTypeSemantics="identifer"
         elementId="95" applicability="all" status="current">
    <description>
      <paragraph>
         Specifies an Application Tag.
      </paragraph>
    </description>
  </field>

  <field name="applicationName"
         dataType="string"
         group="application"
         elementId="96" applicability="all" status="current">
    <description>
      <paragraph>
         Specifies the name of an application.
      </paragraph>
    </description>
  </field>
```

```
     <field name="applicationCategoryName"
            dataType="string"
            group="application"
            elementId="<to be assigned>" applicability="all"
   status="current">
       <description>
         <paragraph>
            An attribute that provides a first level categorization
            for each Application Tag.
         </paragraph>
       </description>
     </field>

     <field name="applicationSubCategoryName"
            dataType="string"
            group="application"
            elementId="<to be assigned>" applicability="all"
   status="current">
       <description>
         <paragraph>
            An attribute that provides a second level
            categorization for each Application Tag.
         </paragraph>
       </description>
     </field>

     <field name="applicationGroupName"
            dataType="string"
            group="application"
            elementId="<to be assigned>" applicability="all"
   status="current">
       <description>
         <paragraph>
              An attribute that groups multiple Application Tags
              that belong to the same networking application.
         </paragraph>
       </description>
     </field>

     <field name="p2pTechnology"
            dataType="string"
            group="application"
            elementId="<to be assigned>" applicability="all"
   status="current">
       <description>
         <paragraph>
```

```
                Specifies if the Application Tag is based on peer-
                to-peer technology. Possible values are: "yes",
                "no", and "unassigned".
          </paragraph>
        </description>
      </field>

      <field name="tunnelTechnology"
             dataType="string"
             group="application"
             elementId="<to be assigned>" applicability="all"
   status="current">
        <description>
          <paragraph>
                   Specifies if the application tag is used as a
                   tunnel technology. Possible values are: "yes",
                   "no", and "unassigned".
          </paragraph>
        </description>
      </field>

      <field name="encryptedTechnology"
             dataType="string"
             group="application"
             elementId="<to be assigned>" applicability="all"
   status="current">
        <description>
          <paragraph>
                   Specifies if the application tag is an encrypted
                   networking protocol. Possible values are: "yes",
                   "no", and "unassigned".
          </paragraph>
        </description>
      </field>
```

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

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

February 14, 2011

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

Abstract

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

Status of this Memo

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

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

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

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

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

This Internet-Draft will expire on April, 2011.

Copyright Notice

Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Table of Contents

1. Introduction

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

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

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

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

1.1. IPFIX Documents Overview

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

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

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

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

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

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

1.2. IPFIX Mediator Documents Overview

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

The "IPFIX Mediation: Framework" [IPFIX-MED-FMWK] provides more architectural details of the arrangement of Intermediate Processes within a Mediator.

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

1. "IP Flow Anonymisation Support", [IPFIX-MED-ANON], which describes anonymisation techniques for IP flow data and the export of anonymised data using the IPFIX protocol.

2. "Flow Selection Techniques" [IPFIX-MED-FLOWSEL], which described the process of selecting a subset of flows from all flows observed at an observation point, along with the motivations, and some specific flow selection techniques.

3. "Exporting Aggregated Flow Data using the IP Flow Information Export" [IPFIX-MED-AGGR] which describes Aggregated Flow export within the framework of IPFIX Mediators and defines an interoperable, implementation-independent method for Aggregated Flow export.

1.3. Relationship with IPFIX and PSAMP

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

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

2. Terminology

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

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

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

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

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

Transport Session Information

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

Original Exporter

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

Original Observation Point

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

IPFIX Mediation

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

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

Intermediate Process

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

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

Intermediate Conversion Process

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

Intermediate Aggregation Process

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

Intermediate Correlation Process

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

Intermediate Selection Process

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

Intermediate Anonymization Process

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

IPFIX Mediator

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

Template Mapping

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

entry in a Template Mapping is scoped by incoming or outgoing Transport Session and Observation Domain, as with Templates and Options Templates in the IPFIX Protocol.

Anonymisation Record

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

Anonymised Data Record

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

Intermediate Anonymisation Process

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

3. Specifications

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

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

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

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

These new specifications, which are more specific compared to [RFC5101], are described with the key words described in [RFC2119].

3.1. Encoding of IPFIX Message Header

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

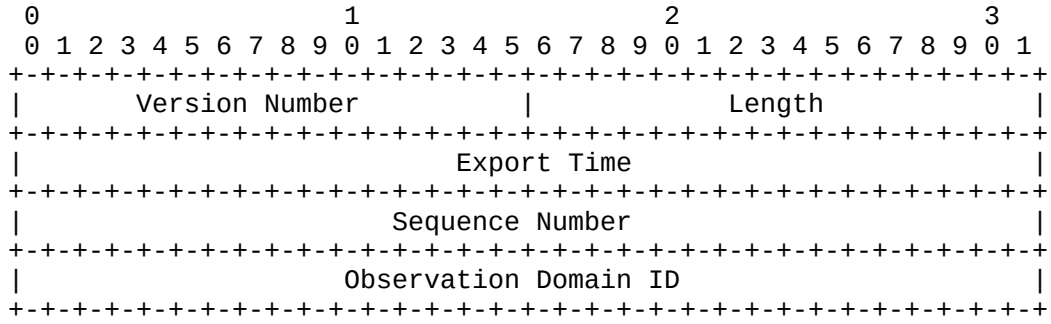


Figure A: IPFIX Message Header format

Message Header Field Descriptions

Version

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

Length

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

Export Time

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

Sequence Number

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

Observation Domain ID

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

3.2. Template Management

3.2.1. Template Management Without Template Records Change

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

managing multiple universities: an unique IPFIX Collector collects all Flow Records for the common infrastructure, but might be re-exporting specific university Flow Records to the responsible system administrator.

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

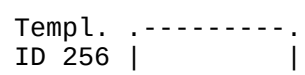
- for each received (Options) Template Record: Template Record Flow Keys and non Flow Keys, Template ID, Observation Domain Id, and Transport Session
- for each exported (Options) Template Record: Template Record Flow Keys and non Flow Keys, Template ID, Collector, Observation Domain Id, and Transport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

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

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

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

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



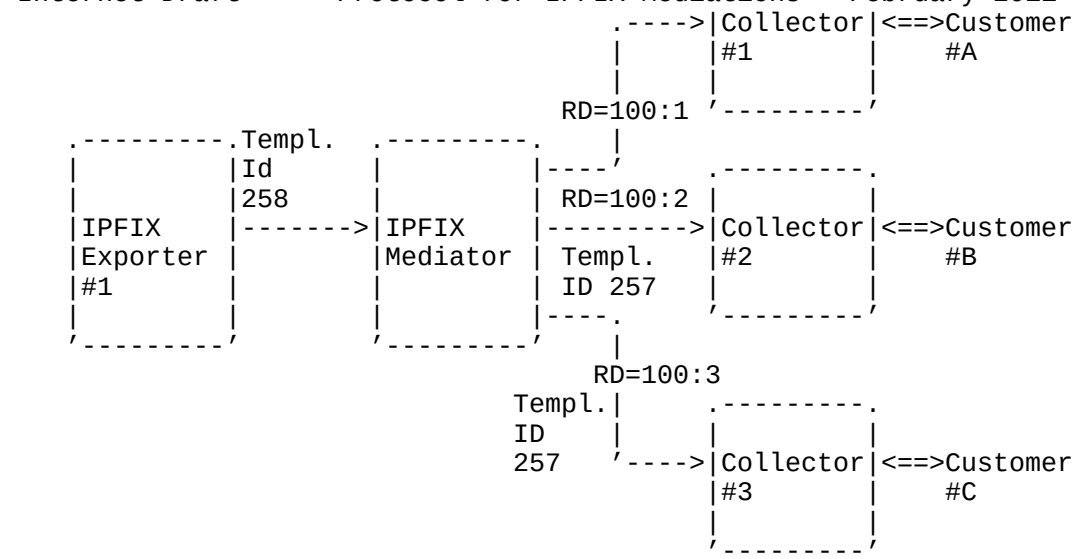


Figure B: Intermediate Aggregation Process Example

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

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

Template Entry C:
 Outgoing Transport Session (to Collector#2):
```

```
                Source IP: <IPFIX Mediator IP address>
                Destination IP: <IPFIX Collector#2 IP address>
      Source Port: <source port>
      Destination Port: 4739 (IPFIX)
    Observation Domain Id: <Observation Domain ID>
    Template Id: 257
      Flow Keys: <series of Flow Keys>
      Non Flow Keys: <series of non Flow Keys>

   Template Entry D:
    Outgoing Transport Session (to Collector#3):
      Source IP: <IPFIX Mediator IP address>
                Destination IP: <IPFIX Collector#3 IP address>
                Source Port: <source port>
      Destination Port: 4739 (IPFIX)
    Observation Domain Id: <Observation Domain ID>
      Template Id: 257
      Flow Keys: <series of Flow Keys>
      Non Flow Keys: <series of non Flow Keys>
```

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

```
   Template Entry A   <----> Template Entry B
   Template Entry A   <----> Template Entry C
   Template Entry A   <----> Template Entry D
```

3.2.2. Template Management With New Template Records

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

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

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

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

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

The following two examples illustrate this.

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

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

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

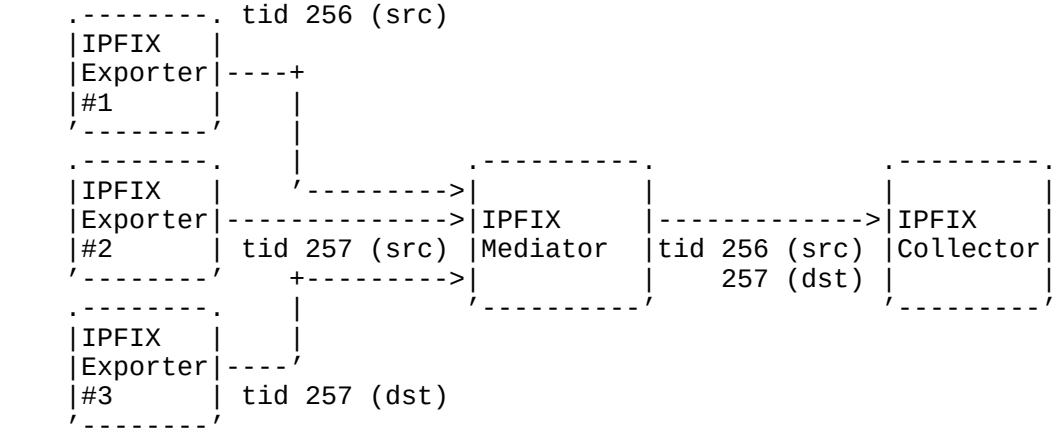


Figure C: Intermediate Aggregation Process Example

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

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

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

```
   Template Entry C:
    Incoming Transport Session (from Exporter#3):
                Source IP: <Exporter#3 export IP address>
                Destination IP: <IPFIX Mediator IP address>
                Source Port: <source port>
      Destination Port: 4739 (IPFIX)
    Observation Domain Id: <Observation Domain ID>
    Template Id: 257
      Flow Keys: destinationIPv4Address
      Non Flow Keys: octetDeltaCount, [others]

   Template Entry D:
    Outgoing Transport Session (to IPFIX Collector):
      Source IP: <IPFIX Mediator export IP address>
      Destination IP: <IPFIX Collector IP address>
      Source Port: <source port>
      Destination Port: 4739 (IPFIX)
    Observation Domain Id: <Observation Domain ID>
      Template Id: 256
      Flow Keys: sourceIPv4Address
      Non Flow Keys: octetDeltaCount

   Template Entry E:
    Outgoing Transport Session (to IPFIX Collector):
      Source IP: <IPFIX Mediator export IP address>
      Destination IP: <IPFIX Collector IP address>
                Source Port: <source port>
      Destination Port: 4739 (IPFIX)
    Observation Domain Id: <Observation Domain ID>
      Template Id: 257
      Flow Keys: destinationIPv4Address
      Non Flow Keys: octetDeltaCount
```

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

```
   Template Entry A   <----> Template Entry D
   Template Entry B   <----> Template Entry D
   Template Entry C   <----> Template Entry E
```

3.3. Time Management

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

of an IPFIX Mediator containing an Intermediate Conversion Process, the IPFIX Mediator MAY keep the export time received from the incoming Transport Session.

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

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

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

3.4. Observation Point Management

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

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

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

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

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

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

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

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

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

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

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

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

3.4.1. Observation Domain Management

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

IPFIX Mediators that do not change (Options) Template Records MUST maintain a Template Mapping, as detailed in section 3.2.1, to ensure that the combination of Observation Domain IDs and Template IDs do not collide on export.

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

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

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

3.5. Specific Reporting Requirements

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

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

implemented. If implemented, the specific Option Templates SHOULD be implemented as specified in these subsections.

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

3.5.1. The Flow Keys Options Template

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

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

templateId	An identifier of a Template. This Information Element MUST be defined as a Scope Field.
flowKeyIndicator	Bitmap with the positions of the Flow Keys in the Data Records.

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

3.5.2. IPFIX Protocol Options Template

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

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

Template", and "The Exporting Process Reliability Statistics Options Template"

3.5.3. IPFIX Mediator Options Template

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

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

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

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

3.6. The Collecting Process’s Side

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

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

3.7. Configuration Management

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

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

4. New Information Elements

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

4.1. - originalExporterIPv4Address

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

Abstract Data Type: ipv4Address

ElementId: TBD3

Status: Proposed

4.2. originalExporterIPv6Address

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

Abstract Data Type: ipv6Address

ElementId: TBD2

Status: Proposed

4.3. originalObservationDomainId

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

Abstract Data Type: unsigned32

ElementId: TBD3

Status: Proposed

5. Security Considerations

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

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

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

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

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

6. IANA Considerations

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

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

EDITOR'S NOTE: the XML specification in Appendix A must be updated with the elementID values allocated, i.e. TBD1, TBD2, and TDB3, must be replaced.

6.1. originalExporterIPv4Address

Name: originalExporterIPv4Address
Description:
The IPv4 address used by the Exporting Process on the Original Exporter. This is used by an IPFIX Mediator Exporting Process to identify the Original Exporter.
Abstract Data Type: ipv4Address
Data Type Semantics: identifier
ElementId: TBD1
Status: current

6.2. originalExporterIPv6Address

Name: originalExporterIPv6Address
Description:
The IPv6 address used by the Exporting Process on the Original Exporter. This is used by the IPFIX Mediator Exporting Process to identify the Original Exporter.
Abstract Data Type: ipv6Address
Data Type Semantics: identifier
ElementId: TBD2
Status: current

6.3. originalObservationDomainId

Name: originalObservationDomainId
Description:
An identifier of the Observation Domain on the Original Exporter, where the metered IP packets are observed. This is used by the IPFIX Mediator Exporting Process to identify an Observation Domain as received from the Original Exporter.
Abstract Data Type: unsigned32
Data Type Semantics: identifier
ElementId: TBD3
Status: current

7. References

7.1. Normative References

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

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

[RFC4960] Stewart, R., Ed., "Stream Control Transmission Protocol", RFC 4960, September 2007.

[RFC5101] Claise, B., Ed., "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information", RFC 5101, January 2008.

[RFC5102] Quittek, J., Bryant, S., Claise, B., Aitken, P., and J. Meyer, "Information Model for IP Flow Information Export", RFC 5102, January 2008.

[RFC5655] Trammell, B., Boschi, E., Mark, L., Zseby, T., and A. Wagner, "Specification of the IP Flow Information Export (IPFIX) File Format", RFC 5655, October 2009.

[RFC5815] Dietz, T., Kobayashi, A., Claise, B., and G. Muenz, "Definitions of Managed Objects for IP Flow Information Export", RFC 5815, April 2010.

[IPFIX-MED-FLOWSEL] Peluso, L., Zseby, T., D'antonio, S., and M. Molina, "Flow Selection Techniques", draft-ietf-ipfix-

flow-selection-tech-04.txt, Internet-Draft work in progress, January 2011.

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

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

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

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

7.2. Informative References

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

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

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

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

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

[RFC5472] Zseby, T., Boschi, E., Brownlee, N., and B. Claise, "IP Flow Information Export (IPFIX) Applicability", RFC 5472, March 2009

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

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

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

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

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

8. Author's Addresses

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

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

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

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

Brian Trammell
ETH Zurich
Gloriastrasse 35
8092 Zurich
Switzerland

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

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

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

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

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


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

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

IPFIX Working Group
Internet-Draft
Intended Status: Standards Track
Expires: April 16, 2011

A. Johnson
B. Claise
P. Aitken
Cisco Systems, Inc.
October 16, 2010

Exporting MIB variables using the IPFIX Protocol
draft-johnson-ipfix-mib-variable-export-00

Abstract

This document specifies a way to export Management Information Base (MIB) objects within the IPFIX protocol, avoiding the need to define new IPFIX Information Elements for existing Management Information Base objects that are already fully specified.

This method requires an extension to the current IPFIX protocol. New Template Set and Options Template Sets are specified to allow the export of Simple Network Management Protocol (SNMP) MIB Objects.

Copyright Notice

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

TO DO

ACTION: need a third example with an Options Template Set to show TBD2 in action. TBD2 is the Set ID of an Options Template Set that uses the extended Field Specifier.

1. Introduction

There is growing interest in using IPFIX to export management information, especially since periodically exporting large chunks of repetitive data from a device is often more appropriate than using a polling mechanism.

While initially targeted at different problems, there is a large parallel between the information transported via IPFIX and SNMP.

Furthermore, certain Management Information Base objects are highly relevant to flows as they are understood today.

For example, in the IPFIX information model [RFC5102], several Information Elements coming from the SNMP world have already been specified. e.g. the interface's SNMP index (ifIndex, [RFC2863]), the interface's name (ifName,[RFC2863]) and the system uptime (sysUpTime, [RFC3418]). Rather than map existing SNMP MIB Object Identifiers to IPFIX Information Elements on a case by case basis, causing replication between the two models, it would be advantageous to enable the export of any existing or future SNMP objects as part of an IPFIX Data Record. However, this is not possible using the existing IPFIX Template Sets as specified in [RFC5101].

Another advantage of exporting MIB objects via IPFIX is that IPFIX would benefit from an extended series of types to be exported. Indeed, the simple and application-wide data types specified in SMIv2 [RFC2578], along with a new textual conventions, can be exported within IPFIX and then decoded in the Collector.

Yet another advantage of exporting MIB objects via IPFIX is the synchronization of the Data Record counters and the MIB counters. For example, if a Network Management Station (NMS) needs the interface counters, coming from ifTable in "The Interfaces Group MIB" [RFC2863]), at the time the Data Record ends, the NMS must poll the interface counters after receiving the Data Records. Unless synchronized Data Records export and SNMP polling is in place (which is difficult/impossible in practice because the Flow duration can not be predicted), the Flow counters and interface counters cannot be compared. With the export of the interface counters within the Data Record, this problem is avoided.

In this document, new Template Sets for Data Records and Options Records are specified to allow Templates to contain any combination of fields defined by traditional IPFIX Information Elements and/or MIB Object Identifiers. The MIB Object Identifiers can reference either non-indexed or indexed MIB objects. When an indexed MIB object is exported, a method to identify how that MIB object was indexed is specified so that the full meaning of the information being exported can be conveyed. A set of example use cases is used to illustrate how these specifications can be used.

Since IPFIX is a push mechanism, initiated from the Exporter with no acknowledgment method, this specification doesn't provide the ability to execute configuration, unlike the SNMP protocol. Instead, this specification allows adding the value of MIB objects into IPFIX Data Records.

2. Terminology

IPFIX-specific terminology used in this document is defined in section 2 of [RFC5101]. For example: Information Element, Template, Template Record, Options Template Record, Template Set, Collector, Exporter, Flow Record, etc... As in [RFC5101], these IPFIX-specific terms have the first letter of a word capitalized.

This document prefers the more generic term "Data Record" as opposed to "Flow Record" as this specification allows the export of MIB objects.

MIB Object Identifier (MIB OID)

> AlphaNumeric-format variable name, denoting a variable name expressed as a sequence of decimal numbers or names separated by periods, as specified by the OBJECT IDENTIFIER in [RFC2578].

MIB Object Identifier Information Element

> An IPFIX Information Element ("MIBObjectIdentifierMarker") that denotes that a MIB Object Identifier is exported in the (Options) Template Record.

3. Example Use Cases

3.1 Detailing CPU Load History

The CPU Usage of a remote network device could be monitored by configuring it to periodically send CPU usage information to a centralized Collector. In this example, the Exporter would send an IPFIX Message every 30 minutes that contained Data Records detailing the CPU 1 minute busy average at 1 minute intervals.

The table of data that is to be exported would look like:

TIMESTAMP	CPU BUSY PERCENTAGE
StartTime + 0 seconds	10%
StartTime + 60 seconds	14%
StartTime + 120 seconds	19%
StartTime + 180 seconds	16%
StartTime + 240 seconds	23%
StartTime + 300 seconds	29%
...	...

The Template Record for such a Data Record will detail two Information Elements:

flowStartSeconds from [RFC5102]:
- Value: IE = 150
- Description: The absolute timestamp of the first packet of this Flow.

cpmCPUTotal1minRev from the proprietary CISCO-PROCESS-MIB
- Value: MIB OID = "1.3.6.1.4.1.9.9.109.1.1.1.1.7"
- Description: The overall CPU busy percentage in the last one-minute period

3.2 Output Interface Queue Size in PSAMP Packet Report

If a PSAMP Packet Report [RFC5476] was generated on any dropped packets on an interface then it may be desirable to know if the send queue on the output interface was full. This could be done be sending the size of the send queue in the same Data Record as the PSAMP Packet Report.

The exported data could look something like:

SRC ADDR	DST ADDR	PAK LEN	OUTPUT I/F	OUTPUT Q. SIZE
192.0.2.1	192.0.2.3	150	Eth 1/0	45
192.0.2.4	192.0.2.9	350	Eth 1/0	45
192.0.2.3	192.0.2.9	650	Eth 1/0	23
192.0.2.4	192.0.2.6	350	Eth 1/1	0

The MIB object for the Output Queue Length, ifOutQLen ("1.3.6.1.2.1.2.2.1.21"), is indexed by the ifIndex interface index as detailed in the IF-MIB [RFC2863]. If, for example, the interface index of "Eth 1/0" in our example is 15, the full MIB Object Identifier for the Output Queue Length would be

"1.3.6.1.2.1.2.2.1.21.15". With the specification in this document, each time a different MIB OID is specified in the Template Record, a new MIB object must be identified, hence a new Template Record. Rather than send a separate Template Record for each Interface Index, it would be much more convenient to identify the index in the Data Record itself.

In fact, only how the indexed object was indexed is important. In our example we identify the Egress Interface, although for other uses it may be sufficient to know that the Output Queue Size was taken from the interface that the packet was switched out of without identifying the actual interface.

The Template Record for our example Data Record would contain the following Information Elements:

```
sourceIPv4Address
destinationIPv4Address
totalLengthIPv4
egressInterface
outboundQueueLength indexed by: egressInterface
```

4. MIB OID Extended Template Formats

Extended Template Record Formats are required to send data defined by MIB Object Identifiers. New Template Sets are required for these extended Template Record Formats.

4.1 MIB OID Extended Template Record Format

The format of the MIB Object Identifier Extended Template Record is shown in Figure A. It consists of a Template Record Header and one or more Field Specifiers.

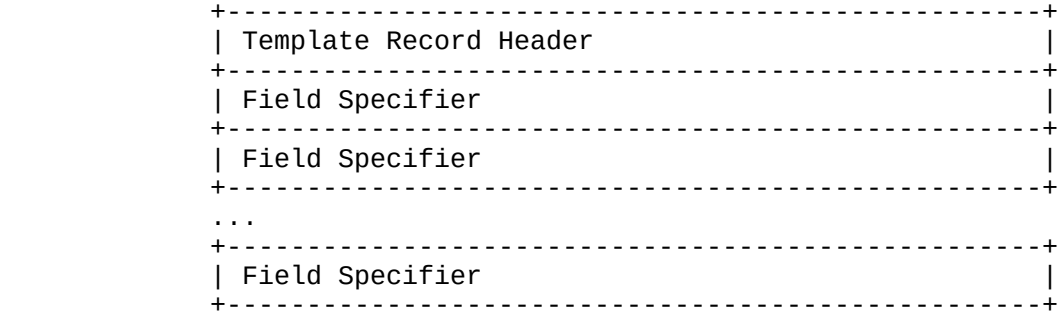


Figure A: MIB Object Identifier Extended Template Record Format

A MIB Object Identifier Extended Template Record MUST contain at least one MIB Object Identifier Extended Field Specifier. It MAY also contain any combination of IANA-assigned and/or Enterprise-Specific Information Element identifiers as specified in [RFC5101].

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

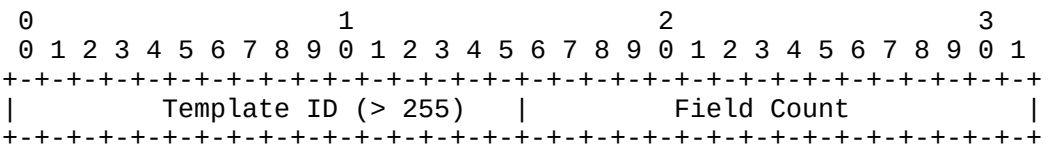


Figure B: Template Record Header Format

Where:

Template ID

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

Field Count

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

At this level of detail the layout of the Template Record Format, as specified in [RFC5101], and the MIB Object Identifier Extended Template Record Format are identical. It is only the structure of the Field Specifiers that is different (see section 4.3).

4.2 MIB OID Extended Options Template Record Format

The format of the MIB Object Identifier Extended Options Template Record is shown in Figure C. It consists of an Options Template Record Header and one or more Field Specifiers.

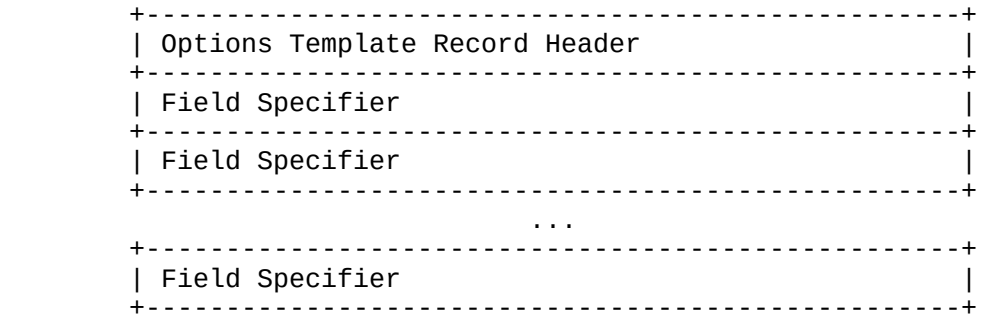


Figure C: MIB Object Identifier Options Extended Template Record Format

A MIB Object Identifier Extended Options Template Record MUST contain at least one MIB Object Identifier Extended Field Specifier, which MAY be a scope field. It MAY also contain any combination of IANA-assigned and/or Enterprise-Specific Information Element identifiers.

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

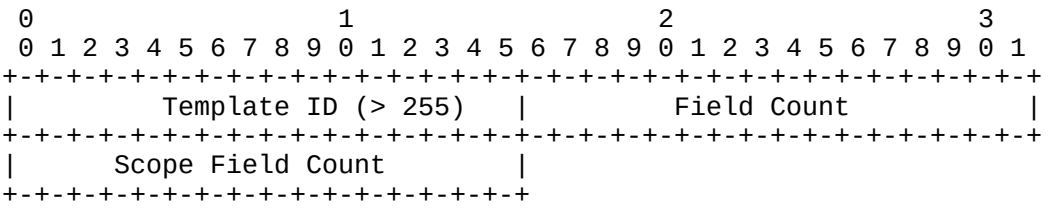


Figure D: Options Template Record Header Format

Where:

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 for an Options Template Record.

As with the Template Record Format, the only difference between the standard Options Template Record Format as defined in [RFC5101] and the MIB Object Identifier Extended Template Options Record Format is the structure of the Field Specifier (see section 4.3).

4.3 MIB OID Extended Field Specifier Format

This section specifies how the Field Specifier format in [RFC5101] is extended to allow fields to be defined using a specified MIB Object. First for a MIB Object Identifier that is a non-indexed MIB object, then for an indexed MIB object.

The Field Specifier formats are shown in Figures E to G below.

4.3.1 Standard Field Specifier Format

The Field Specifier format in figure E, along with the associated definitions, has been copied from [RFC5101], for an easier comparison with the MIB Object Identifier Extended Field Specifier Format in figures F and G.

When sending an IANA-assigned and/or Enterprise-Specific Information Element identifier, the Field Specifier Format is the same as shown below.

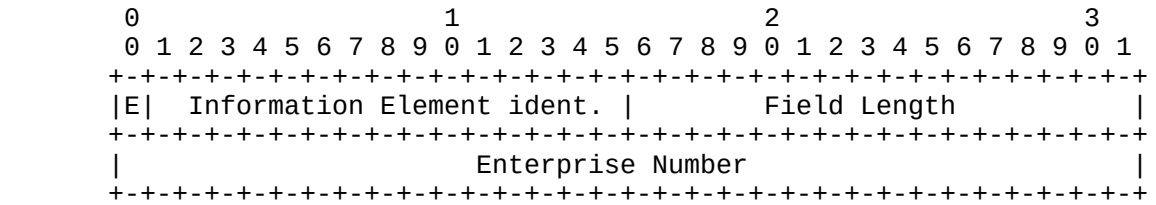


Figure E: Standard Field Specifier format

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 [RFC5102].

Field Length

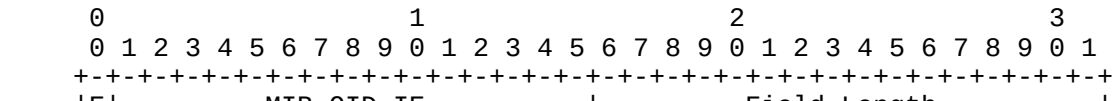
The length of the corresponding encoded Information Element, in octets. Refer to [RFC5102]. The field length may be smaller than the definition in [RFC5102] if reduced size encoding is used (see section 6.2). The value 65535 is reserved for variable length Information Element (see section 7).

Enterprise Number

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

4.3.2 Extended Field Specifier Format for non-indexed MIB Object

When a MIB object is to be exported, a special Information Element value is used to show that the extended Field Specifier is being used, as shown in Figure F:



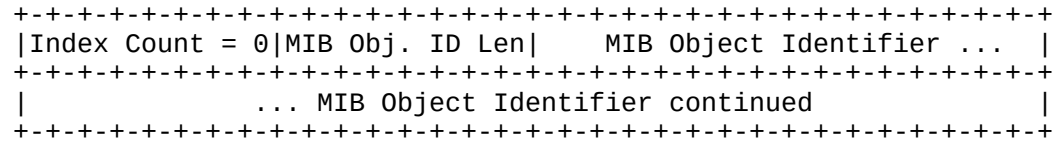


Figure F: MIB Object Identifier Extended Field Specifier Format for a non-indexed MIB Object with an OID length < 255

Where:

E

Enterprise bit. In the special case of a MIB Object Identifier export, the Enterprise bit MUST always be 0, even if the exported MIB object is specified in a proprietary MIB, therefore containing the private enterprise number in its OID.

MIB OID IE

Special IPFIX Information Element, MIBObjectIdentifierMarker, that denotes that a MIB object is exported in the (Options) Template Record. When the MIB Object Identifier Information Element (MIB OID IE) is used, the MIB Object Identifier must be specified in the Field Specifier for the Collecting Process to be able to decode the Records.

Field Length

The definition is as [RFC5101].

Index Count

The number of indexes for a MIB object, and zero for a non-indexed MIB object.

MIB Object Identifier Length

The length of the MIB Object Identifier that follows. This is encoded in the same manner as the variable length encoding in [RFC5101]. If the length of the MIB Object Identifier is greater than or equal to 255 octets, the length is encoded into 3 octets before the MIB Object Name.

The first octet is 255 and the length is carried in the second and third octets (as shown in Figure H).

MIB Object Identifier

An alphanumeric-format variable name which denotes a variable name expressed as a sequence of decimal numbers or names separated by periods, as specified by the OBJECT IDENTIFIER in [RFC2578].

If the MIB Object Identifier is longer than 254 characters then the length MUST be extended:

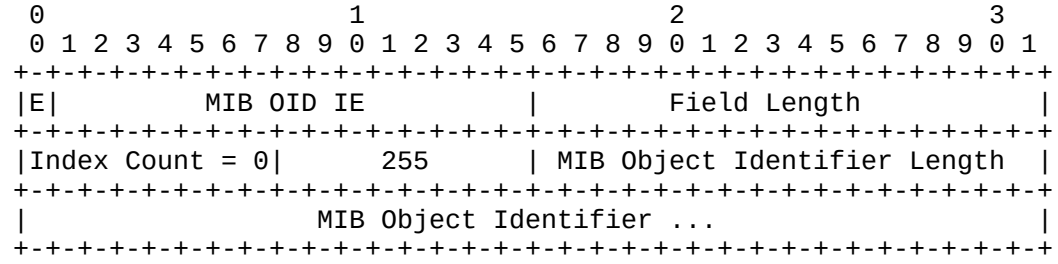
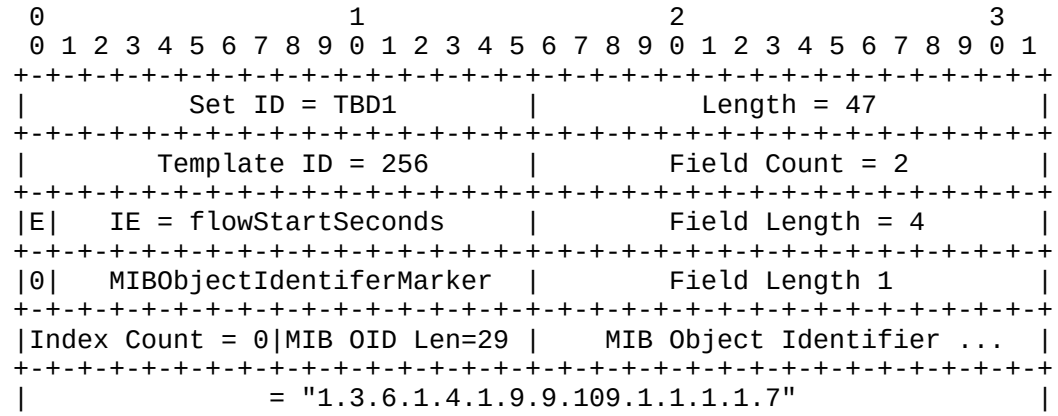


Figure G: MIB Object Identifier Extended Field Specifier Format with OID length >= 255

Figure H shows the exported Template Set detailing the Template Record for exporting CPU Load (see section 3.1).



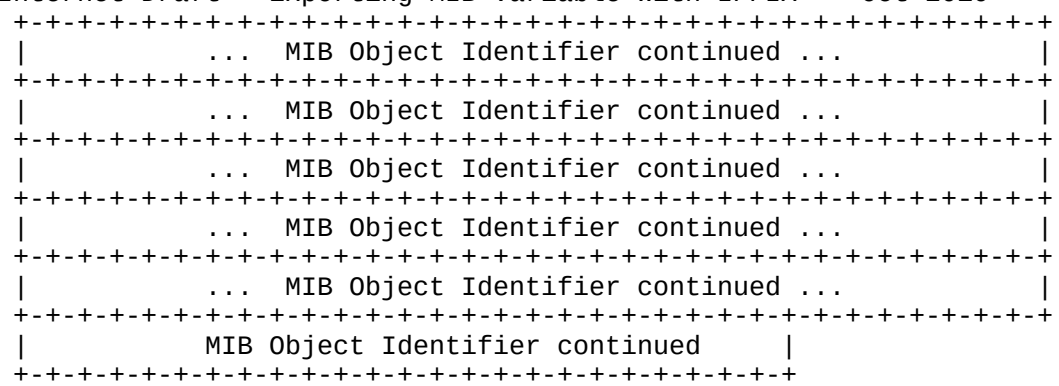


Figure H: Example of CPU Load Template Set

4.3.3 Extended Field Specifier Format for Indexed MIB Object

When an indexed MIB object is exported in IPFIX, the meaning of the exported value each index SHOULD be identified. This index (or indexes) MAY be an IPFIX Information Element or MIB Object Identifier.. Note that the IPFIX Information Element MAY be an enterprise-specific Information Element.

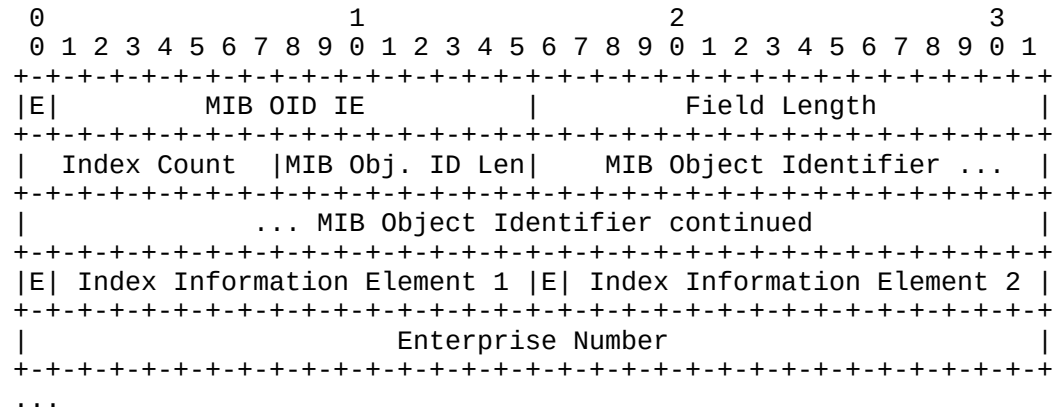


Figure I: MIB Object Identifier Extended Field Specifier Format with an indexed MIB Object using a IPFIX I.E. as Index

Where:

E

Enterprise bit. In the special case of a MIB Object Identifier export, the Enterprise bit MUST always be 0, even if the exported MIB object is specified in a proprietary MIB, therefore containing the private enterprise number in its OID. For any indexes identified using Information Elements the Enterprise bit can be 1, indicating that an Enterprise Number will follow the Information Element.

MIB OID IE

Special IPFIX Information Element, MIBObjectIdentifierMarker, that denotes that a MIB object is exported in the (Options) Template Record. When the MIB Object Identifier Information Element (MIB OID IE) is used, the MIB Object Identifier must be specified in the Field Specifier for the Collecting Process to be able to decode the Records.

Field Length

The definition is as [RFC5101].

Index Count

The number of indexes for a MIB object, and zero for a non-indexed MIB object.

MIB Object Identifier Length

The length of the MIB Object Identifier that follows. This is encoded in the same manner as the variable length encoding in [RFC5101]. If the length of the MIB Object Identifier is greater than or equal to 255 octets, the length is encoded into 3 octets before the MIB Object Name. The first octet is 255 and the length is carried in the second and third octets (as shown in Figure H).

MIB Object Identifier

An alphanumeric-format variable name which denotes a variable name expressed as a sequence of decimal numbers or names separated by periods, as specified by the OBJECT IDENTIFIER in [RFC2578].

A MIB Object Identifier MAY be used as an index and sent as described in Figure J. If a MIB Object Identifier with an index is used as an index then its indexes will no be identified.

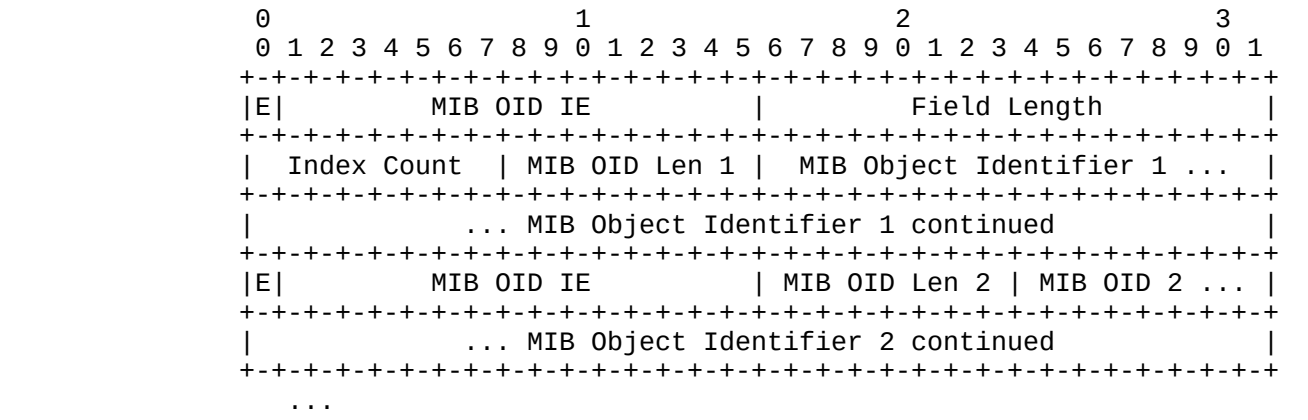


Figure J: MIB Object Identifier Extended Field Specifier Format with a MIB Index using a normal MIB Object Identifier as index

Where:

E

Enterprise bit. In the special case of a MIB Object Identifier export, the Enterprise bit MUST always be 0, even if the exported MIB object is specified in a proprietary MIB, therefore containing the private enterprise number in its OID.

MIB OID IE

Special IPFIX Information Element, MIBObjectIdentifierMarker, that denotes that a MIB object is exported in the (Options) Template Record. When the MIB Object Identifier Information Element (MIB OID IE) is used, the MIB Object Identifier must be specified in the Field Specifier for the Collecting Process to be able to decode the Records.

Field Length

The definition is as [RFC5101].

Index Count

The number of indexes for a MIB object, and zero for a non-indexed MIB object.

MIB Object Identifier Length 1

The length of the MIB Object Identifier being exported. This is encoded in the same manner as the variable length encoding in [RFC5101]. If the length of the MIB Object Identifier is greater than or equal to 255 octets, the length is encoded into 3 octets before the MIB Object Name. The first octet is 255 and the length is carried in the second and third octets.

MIB Object Identifier 1

An alphanumeric-format variable name of the MIB Object Identidier being exported, which denotes a variable name expressed as a sequence of decimal numbers or names separated by periods, as specified by the OBJECT IDENTIFIER in [RFC2578].

MIB Object Identifier Length 2

The length of the MIB Object Identifier being used as an index. This is encoded in the same manner as the variable length encoding in [RFC5101]. If the length of the MIB Object Identifier is greater than or equal to 255 octets, the length is encoded into 3 octets before the MIB Object Name. The first octet is 255 and the length is carried in the second and third octets.

MIB Object Identifier 1

An alphanumeric-format variable name of the MIB Object Identifier being used as an index, which denotes a variable name expressed as a sequence of decimal numbers or names separated by periods, as specified by the OBJECT IDENTIFIER in [RFC2578].

Figure K shows the exported Template Set detailing the Template for exporting a PSAMP Report with Output Queue Size (see section 3.2).

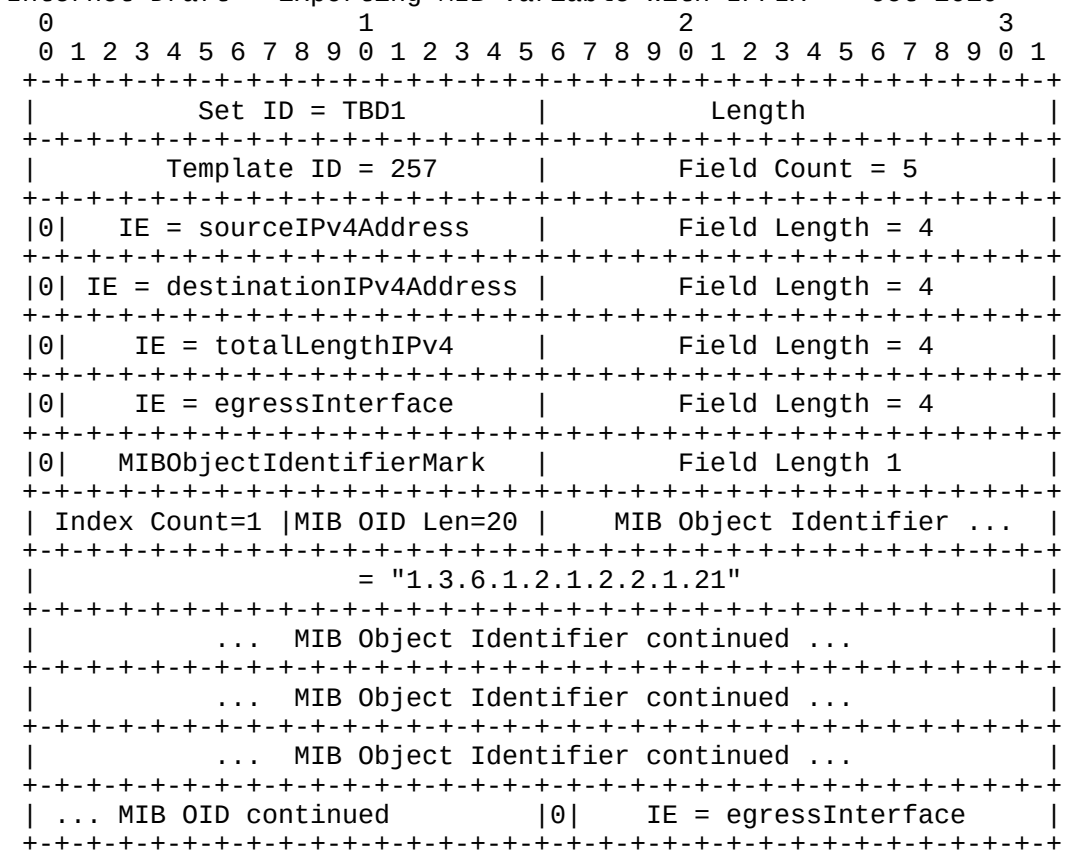
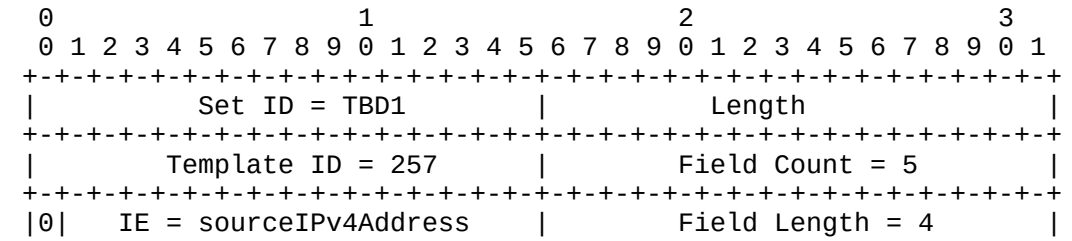


Figure K: Example of PSAMP Report with Output Queue Size

Figure L shows the exported Template Set detailing the Template for exporting a PSAMP Report with Output Queue Size but using the ifIndex MIB object as the exported index, rather than the Egress Interface Information Element.



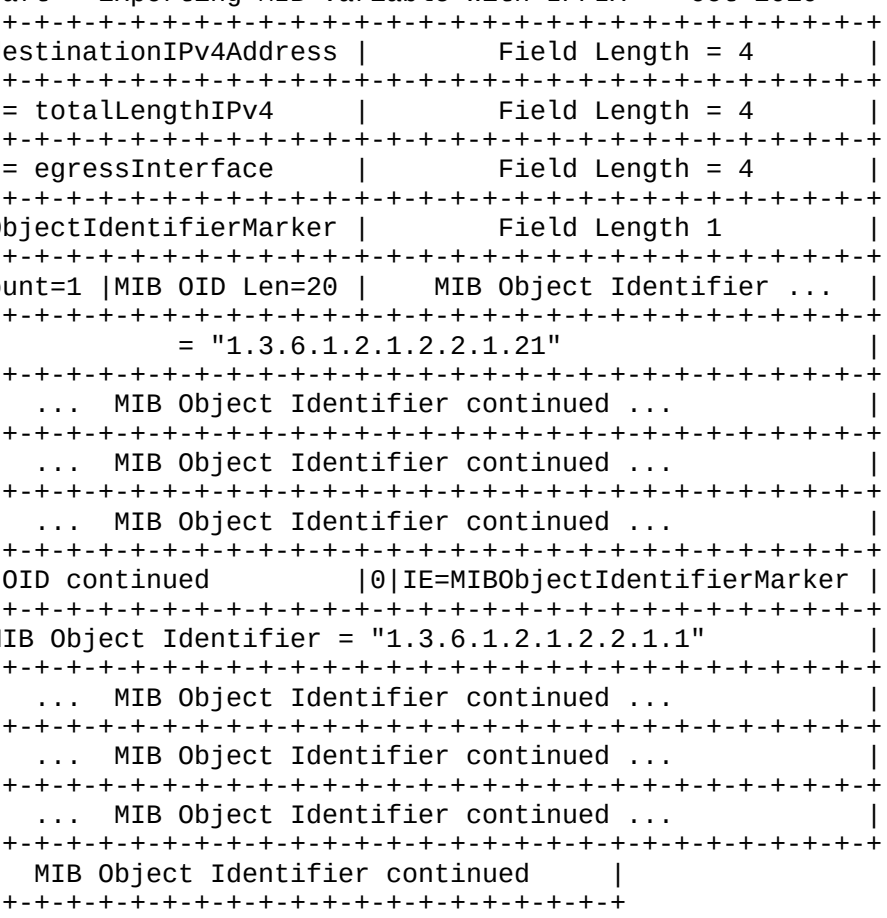


Figure L: Example of PSAMP Report with Output Queue Size using ifIndex from IF-MIB as an index [RFC2578]

4.4 Template Management

Templates are managed as per [RFC5101].

The Set ID field MUST contain the value TBD1 for any Template Set that contains a MIB Object Identifier Extended Field Specifier. The Template Withdrawal Message for such a Template must also use a Set ID field containing the value TBD1.

The Set ID field MUST contain the value TBD2 for any Option Template Set that contains a MIB Object Identifier Extended

Field Specifier. The Template Withdrawal Message for such an Option Template must also use a Set ID field containing the value TBD2.

5. 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 of RFC 5101.

The specifications in section 9 of RFC 5101 also apply to Collector's that implement this specification. In addition, the following specifications should be noted.

A Collecting Process that implements this specification MUST be able to receive Set IDs TBD1 and TBD2, as specified in this document.

A Collecting Process that implements this specification MUST have access to a MIB database in order to look up the received MIB Object IDs and find the type and name of MIB OID fields used in received templates. It should be noted that since reduced length encoding MAY be used by the Exporting Process then the Collecting Process cannot assume a received size for a field is the maximum size it should expect for that field.

If a Collecting Process receives a MIB Object ID that it cannot decode, it SHOULD log an error.

If a Collecting Process receives a MIB Object ID for an indexed MIB Object but isn't sent the appropriate number of indexes then it SHOULD log an error, but it MAY use the Template Record to decode the Data Records as the associated indexes are purely semantic information.

6. Applicability

Making available the many and varied items from the MIBs opens up a wide range of possible applications for the IPFIX protocol, some quite different from the usual flow information. Some potential enhancements for traditional applications are detailed below:

Some monitoring applications periodically export an interface id to interface name mapping using IPFIX Options Templates. This could be expanded to include the MIB Object "1.3.6.1.2.1.2.ifTable.ifEntry.ifInUcastPkts" indexed using the ingressInterface Information Element, as a index. This would give the input statistics for each interface which can be compared to the flow information to ensure the sampling rate is expected. Or, if there is no sampling, to ensure that all the expected packets are being monitored.

7. Security Considerations

For this extension to the IPFIX protocol, the same security considerations as for the IPFIX protocol apply [RFC5101].

However, the Metering Process MUST check whether or not the MIB variables can be accessed, and hence exported with IPFIX. Therefore a read or read-write community string in SNMPv1 and SNMPv2c, or a principal in SNMPv3, MUST be associated with the Metering Process.

If the management entity supports the View-based Access Control Model (VACM) for the SNMP [RFC3415], then the Metering Process MUST validate with the View-Based Access Control [RFC3415] that the MIB object can accessed before exporting his content.

If there is a view in case of SNMPv1 and SNMPv2c, the Metering Process MUST validate that the MIB object can accessed before exporting his content.
Whether the Exporter allows or not the configuration of Template that contains an unauthorized MIB object is implementation specific.

8. 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 previously reserved Set ID values of TBD1 and TBD2 are used as specified in this document. All other Set ID values are reserved for future use. Set ID values above 255 are used for Data Sets.

A new Information Element, "MIBObjectIdentifierMarker", needs to be reserved.

9. References

9.1 Normative References

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

[RFC2578] McCloghrie, K. Perkins, D., Schoenwaelder, J. , Structure of Management Information Version 2 (SMIv2), RFC 2578, April 1999

[RFC2863] McCloghrie, K., Kastenholz, F., Interfaces Group MIB, RFC 2863, June 2000

[RFC3415] Wijnen, B., Presuhn, R., and K. McCloghrie, View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP), RFC3415, December 2002

[RFC3418] Presuhn, R., Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, Management Information Base (MIB) for the Simple Network Management Protocol (SNMP), RFC3418, December 2002

[RFC5101] B. Claise et Al, IPFIX Protocol Specification, RFC 5101, January 2008

[RFC5102] J. Quittek, S. Bryant, B. Claise, J. Meyer, Information Model for IP Flow Information Export, RFC5102, January 2008

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

9.2 Informative References

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

Author's Addresses

Andrew Johnson
Cisco Systems (Scotland) Ltd.
96 Commercial Quay
Commercial Street
Edinburgh, EH6 6LX, United Kingdom
Phone: +44 131 561 3641
Email: andrjohn@cisco.com

Benoit Claise
Cisco Systems
De Kleetlaan 6a b1
Diegem 1813
Belgium
Phone: +32 2 704 5622
Email: bclaise@cisco.com

Paul Aitken
Cisco Systems (Scotland) Ltd.
96 Commercial Quay
Commercial Street
Edinburgh, EH6 6LX, United Kingdom
Phone: +44 131 561 3616
Email: paitken@cisco.com

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

S. Kashima
K. Nakata
NTT
A. Kobayashi
NTT East

March 15, 2011

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

Abstract

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

Table of Contents

1. Introduction

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

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

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

1.1. Conventions Used in This Document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2. Extended Ethernet Technology

2.1. Wide-Area Ethernet Network Summary

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

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

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

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

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

2.2. Data Center Network Summary

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

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

2.3. Multiple Path Ethernet Summary

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

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

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

3. New Information Elements

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

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

3.1. dataLinkFrameSize

Description:

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

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

Abstract Data Type: unsigned16

Data Type Semantics: quantity

ElementId: 312

Status: current

3.2. dataLinkFrameSection

Description:

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

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

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

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

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

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

Abstract Data Type: octetArray

ElementId: 315

Status: current

3.3. dataLinkFrameType

Description:

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

The following data link types are defined here.

- 0x01 ETHERNET

Further values may be assigned by IANA.

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

Abstract Data Type: unsigned16

Data Type Semantics: identifier

ElementId: 347

Status: current

3.4. sectionOffset

Description:

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

Abstract Data Type: unsigned16

Data Type Semantics: quantity

ElementId: 348

Status: current

3.5. sectionObservedOctets

Description:

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

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

Abstract Data Type: unsigned16

Data Type Semantics: quantity

ElementId: 349

Status: current

3.6. dot1qServiceInstanceTag

Description:

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

Abstract Data Type: octetArray

Data Type Semantics: identifier

ElementId: 350

Status: current

3.7. dot1qServiceInstanceId

Description:

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

Abstract Data Type: unsigned32

Data Type Semantics: identifier

ElementId: 351

Status: current

3.8. dot1qServiceInstancePriority

Description:

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

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

Abstract Data Type: unsigned8

Data Type Semantics: identifier

ElementId: 352

Status: current

3.9. dot1qCustomerDestinationMacAddress

Description:

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

Abstract Data Type: macAddress

Data Type Semantics: identifier

ElementId: 353

Status: current

3.10. dot1qCustomerSourceMacAddress

Description:

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

Abstract Data Type: macAddress

Data Type Semantics: identifier

ElementId: 354

Status: current

4. Modification of Existing Information Elements Related to Packet Section

--- This is open issue. ---

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

4.1. ipHeaderPacketSection

Description:

TBD

Abstract Data Type: octetArray

ElementId: 313

Status: current

4.2. ipPayloadPacketSection

Description:

TBD

Abstract Data Type: octetArray

ElementId: 314

Status: current

4.3. mplsLabelStackSection

Description:

TBD

Abstract Data Type: octetArray

ElementId: 316

Status: current

4.4. mplsPayloadPacketSection

Description:

TBD

Abstract Data Type: octetArray

ElementId: 317

Status: current

5. Modification of Existing Information Elements Related to VLAN Tag

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

5.1. dot1qVlanId

Description:

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

Abstract Data Type: unsigned16

Data Type Semantics: identifier

ElementId: 243

Status: current

Reference:

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

5.2. dot1qPriority

Description:

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

Abstract Data Type: unsigned8

Data Type Semantics: identifier

ElementId: 244

Status: current

Reference:

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

5.3. dot1qCustomerVlanId

Description:

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

the C-TAG Tag Control Information (TCI) field in encapsulated IEEE 802.1ad or IEEE 802.1Q frame as described in [IEEE802.1ah-2008]. (*) Note: the 802.1Q tag directly following the outer one.

Abstract Data Type: unsigned16

Data Type Semantics: identifier

ElementId: 245

Status: current

Reference:

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

5.4. dot1qCustomerPriority

Description:

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

Abstract Data Type: unsigned8

Data Type Semantics: identifier

ElementId: 246

Status: current

Reference:

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

(3) [IEEE802.1ah-2008]

6. Security Considerations

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

7. IANA Considerations

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

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

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

8. References

8.1. Normative References

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

8.2. Informative References

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

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

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

[IEEE802.1Q-2005]
IEEE Computer Society, "IEEE Standards for Local and Metropolitan Area Networks: Virtual Bridged Local Area Networks", IEEE Std 802.1Q-2005, May 2006.

[IEEE802.1Qbh]
IEEE Computer Society, "Draft Standards for Local and Metropolitan Area Networks: Virtual Bridged Local Area Networks Amendment: Bridge Port Extension", IEEE Std P802.1Qbh/D1.1, February 2011.

[IEEE802.1ad-2005]
IEEE Computer Society, "IEEE Standards for Local and Metropolitan Area Networks: Virtual Bridged Local Area Networks Amendment 4: Provider Bridges", IEEE Std 802.1ad-2005, May 2006.

[IEEE802.1ah-2008]
IEEE Computer Society, "IEEE Standards for Local and Metropolitan Area Networks: Virtual Bridged Local Area Networks Amendment 7: Provider Backbone Bridges", IEEE Std 802.1ah-2008, August 2008.

[IEEE802.1aq]
IEEE Computer Society, "Draft Standards for Local and Metropolitan Area Networks: Virtual Bridged Local Area Networks Amendment: Shortest Path Bridging", IEEE Std P802.1aq/D3.6, February 2011.

[IEEE802.3-2005]
IEEE Computer Society, "IEEE Standard for Information Technology - Telecommunications and Information Exchange Between Systems - Local and Metropolitan Area Networks - Specific Requirements Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications", IEEE Std 802.3-2005, December 2005.

[ISO_IEC.10589_2002]
International Organization for Standardization, "Information technology -- Telecommunications and information exchange between systems -- Intermediate System to Intermediate System intra-domain routing information exchange protocol for use in conjunction with the protocol for providing the connectionless-mode network service (ISO 8473)", ISO Standard 10589:2002, November 2002.

[ISO_IEC.7498-1_1994]
International Organization for Standardization, "Information technology -- Open Systems Interconnection -- Basic Reference Model: The Basic Mode", ISO Standard 7498-1:1994, June 1996.

[RFC2863] McCloghrie, K. and F. Kastenholz, "The Interfaces Group MIB", RFC 2863, June 2000.

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

[RFC5101] Claise, B., "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information", RFC 5101, January 2008.

[RFC5475] Zseby, T., Molina, M., Duffield, N., Niccolini, S., and F. Raspall, "Sampling and Filtering Techniques for IP Packet Selection", RFC 5475, March 2009.

[RFC5477] Dietz, T., Claise, B., Aitken, P., Dressler, F., and G. Carle, "Information Model for Packet Sampling Exports", RFC 5477, March 2009.

Appendix A. Frame Formats in Wide-Area Ethernet Network

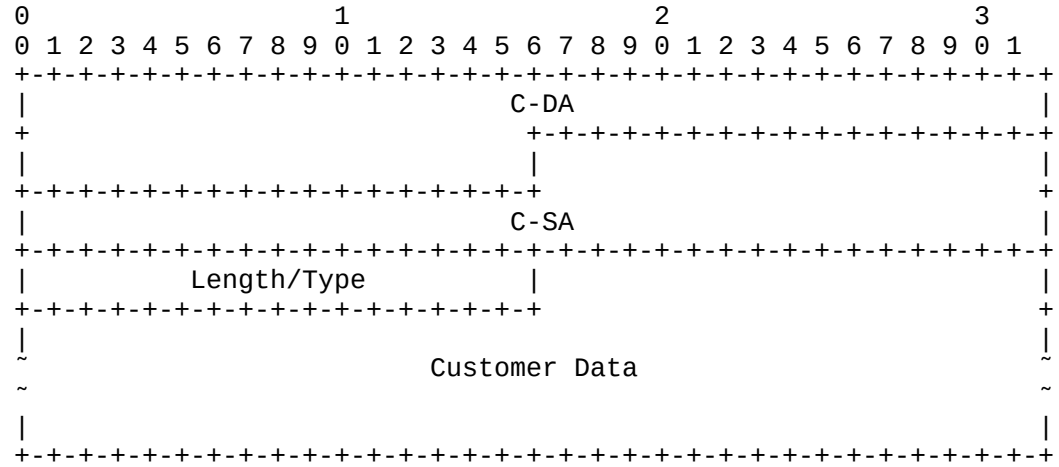


Figure A-1: IEEE802.1D Frame Format in Customer Bridged Network

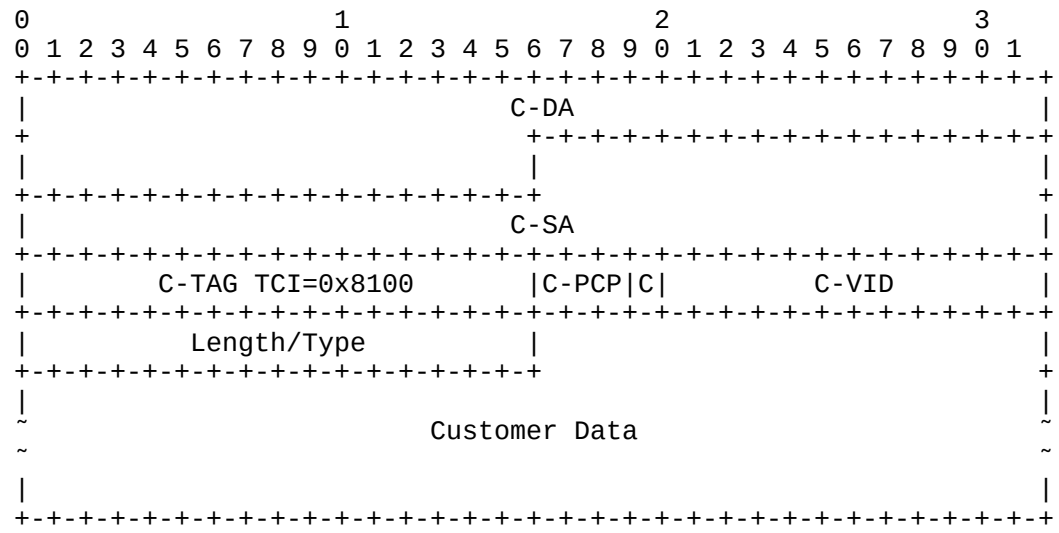


Figure A-2: IEEE802.1Q Frame Format in Customer Bridged Network

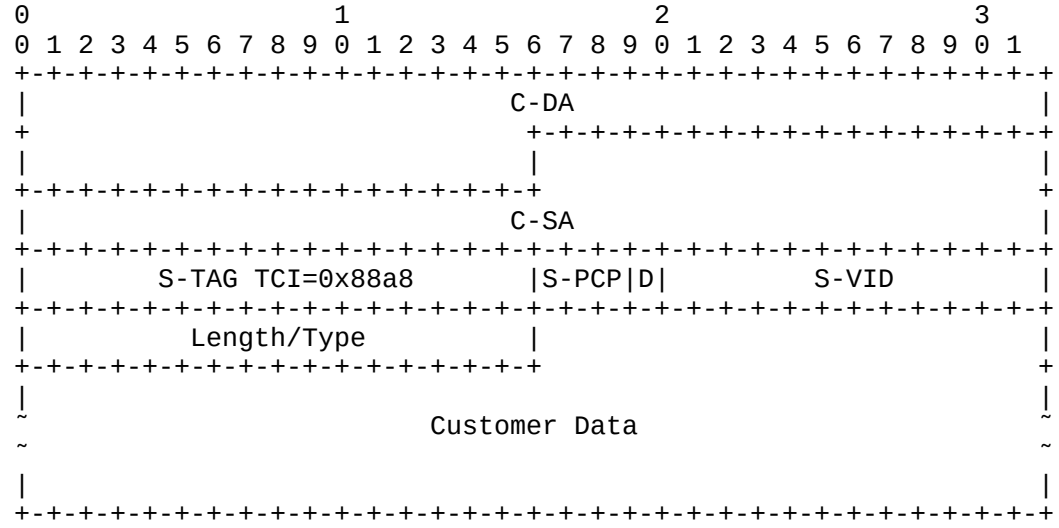


Figure A-3: IEEE802.1ad (no C-Tag) Frame Format in Provider Bridged Network

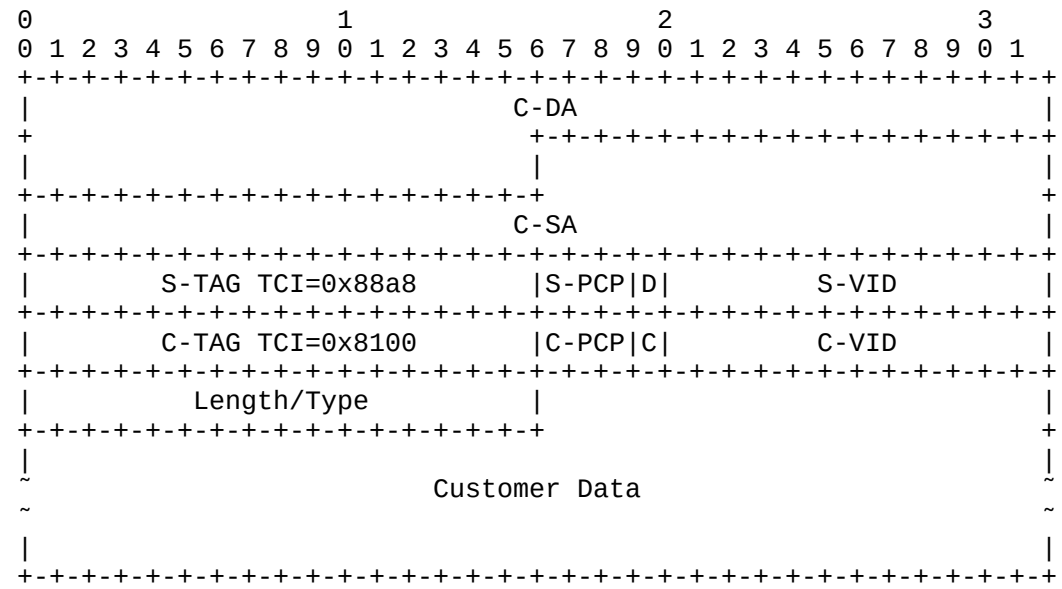


Figure A-4: IEEE802.1ad (C-Tagged) Frame Format in Provider Bridged Network

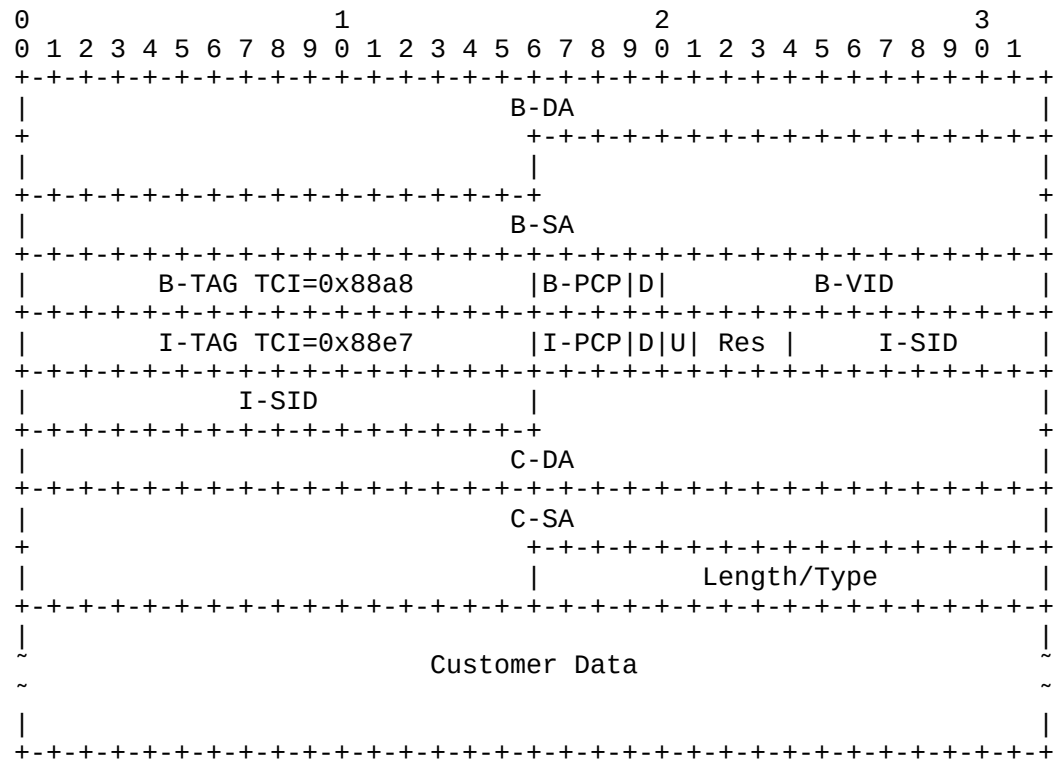


Figure A-5: IEEE802.1ah (no C-Tag) Frame Format in Provider Backbone Bridged Network

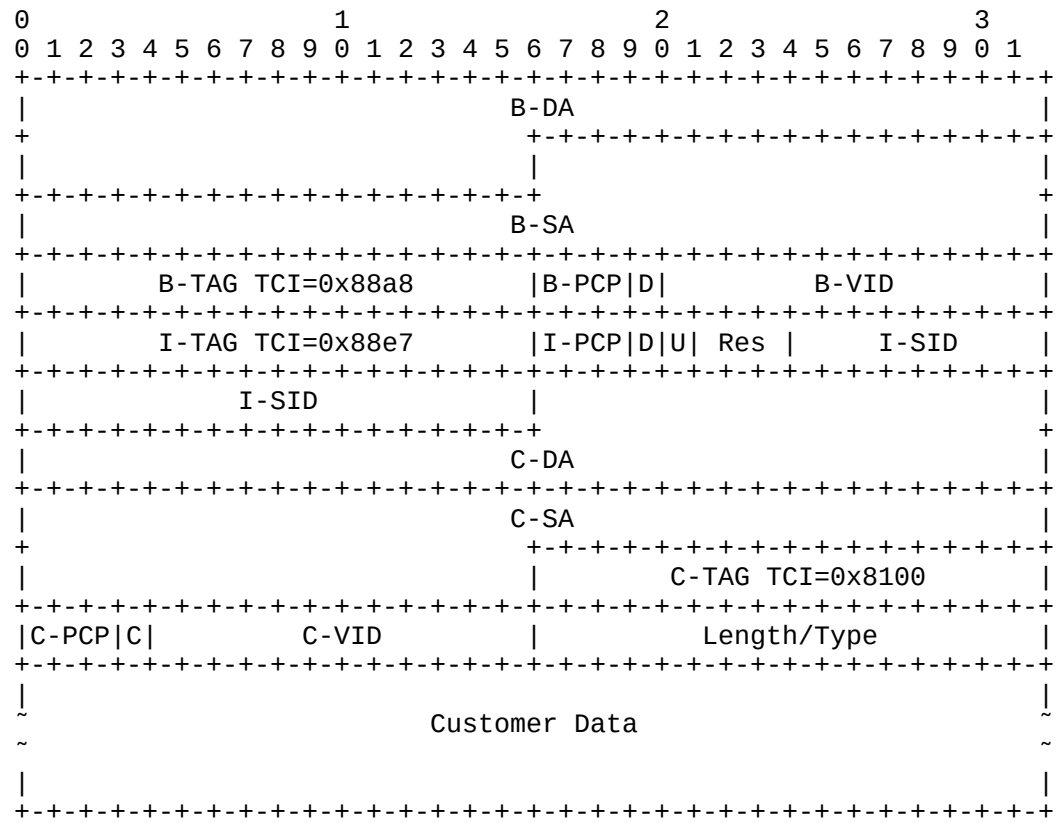


Figure A-6: IEEE802.1ah (C-Tagged) Frame Format in Provider Backbone Bridged Network

Appendix B. Frame Formats in Data Center Network

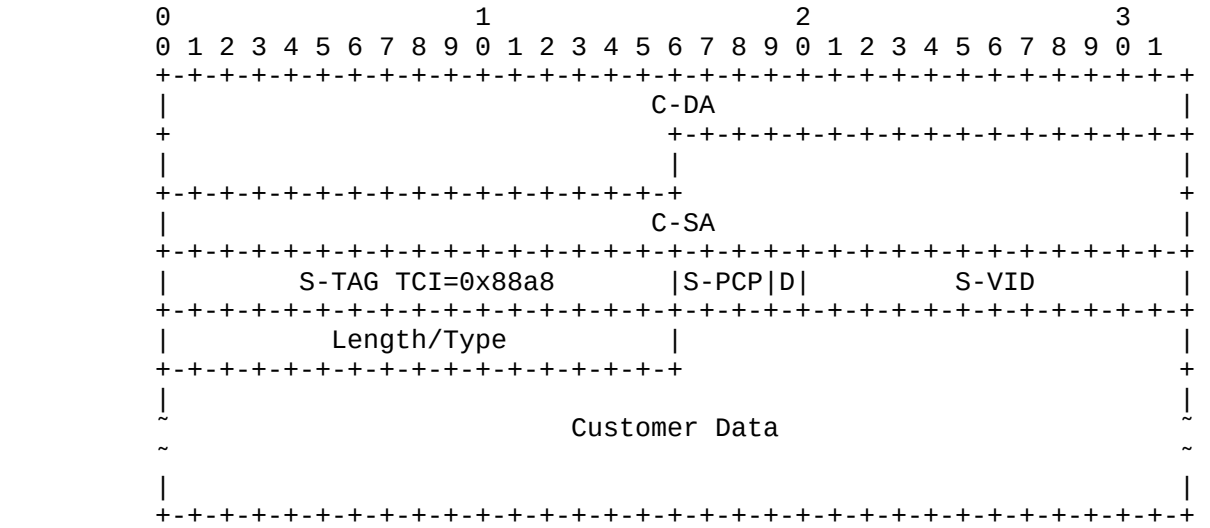


Figure B-1: IEEE802.1Qbh (S-TAG) Frame Format in Data Center Network

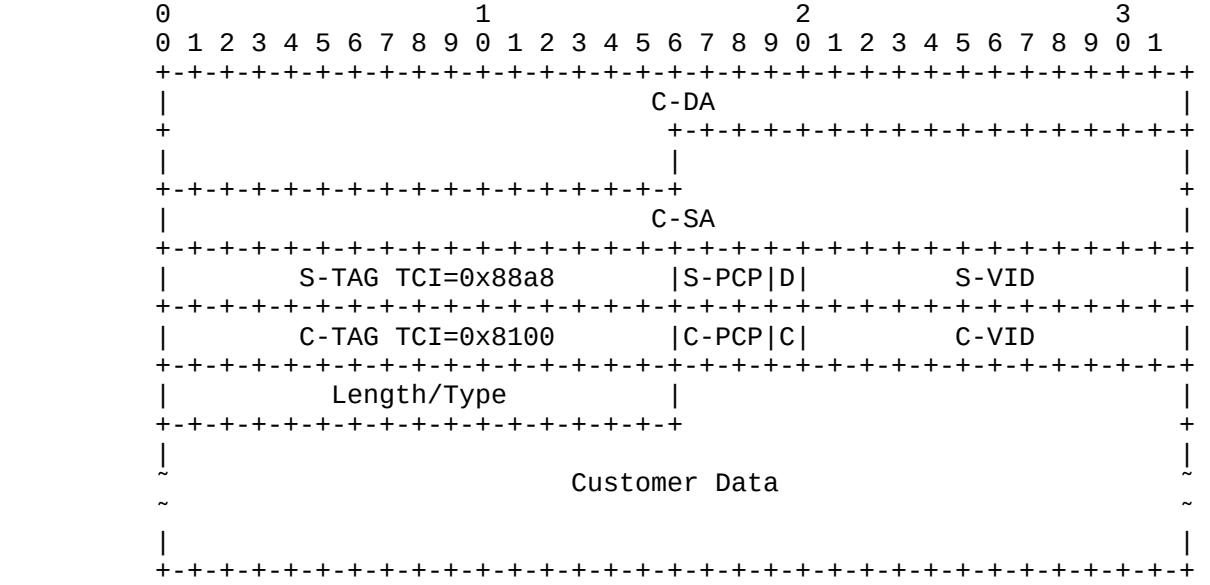


Figure B-2: IEEE802.1Qbh (S-TAG+C-TAG) Frame Format in Data Center Network

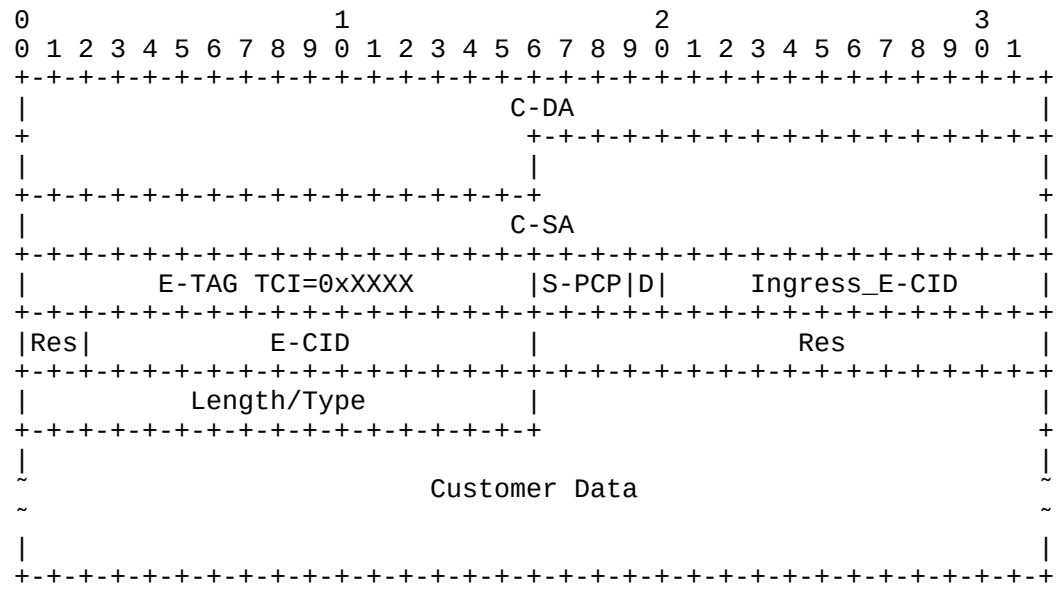


Figure B-3: IEEE802.1Qbh (E-TAG) Frame Format in Data Center Network

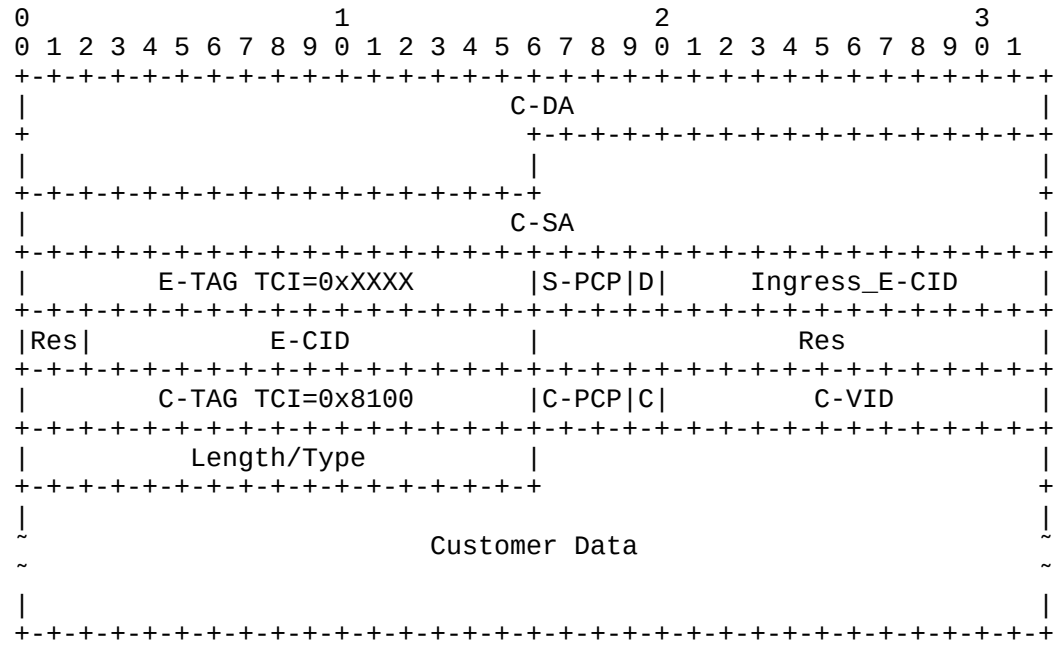


Figure B-4: IEEE802.1Qbh (E-TAG+C-TAG) Frame Format in Data Center

Network

Appendix C. Frame Format in TRILL

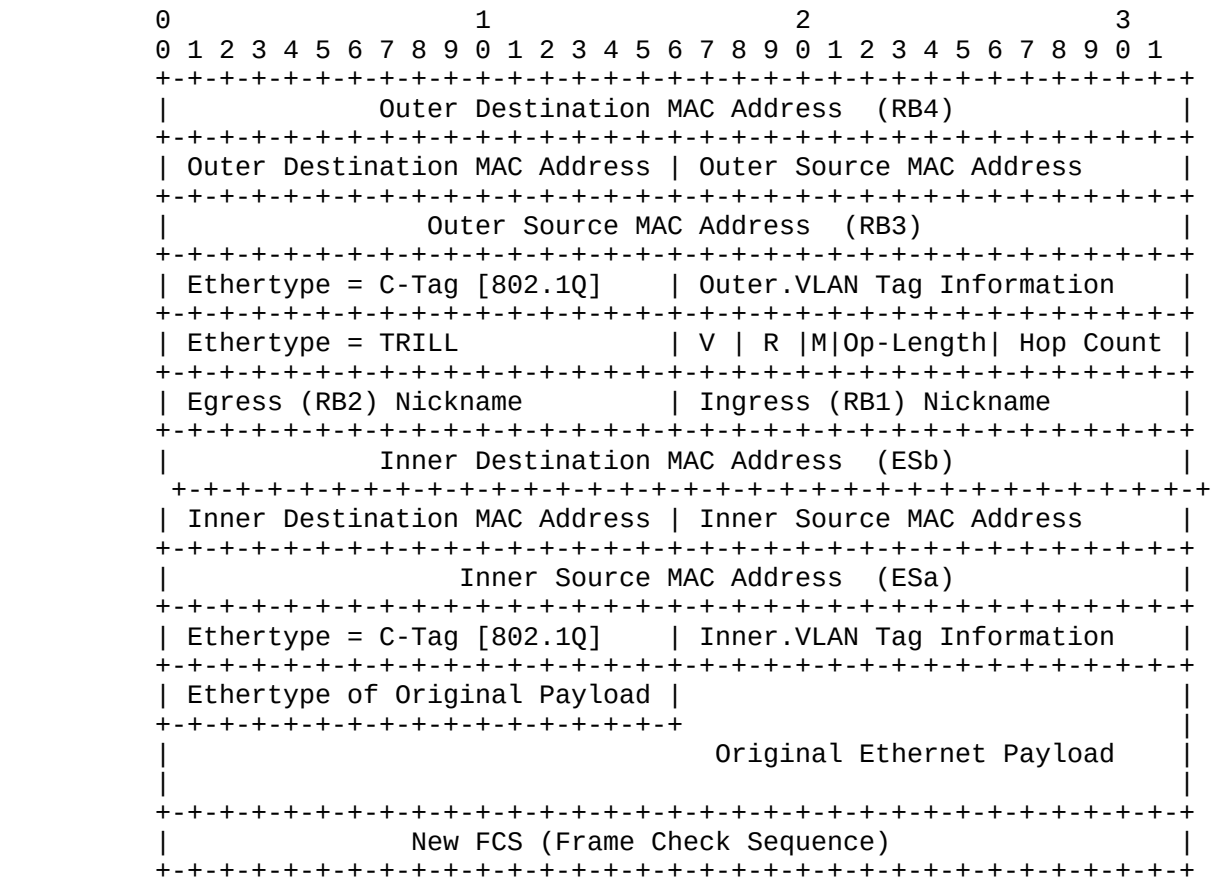


Figure C-1: TRILL Data Encapsulation over Ethernet

Appendix D. Template Formats Example

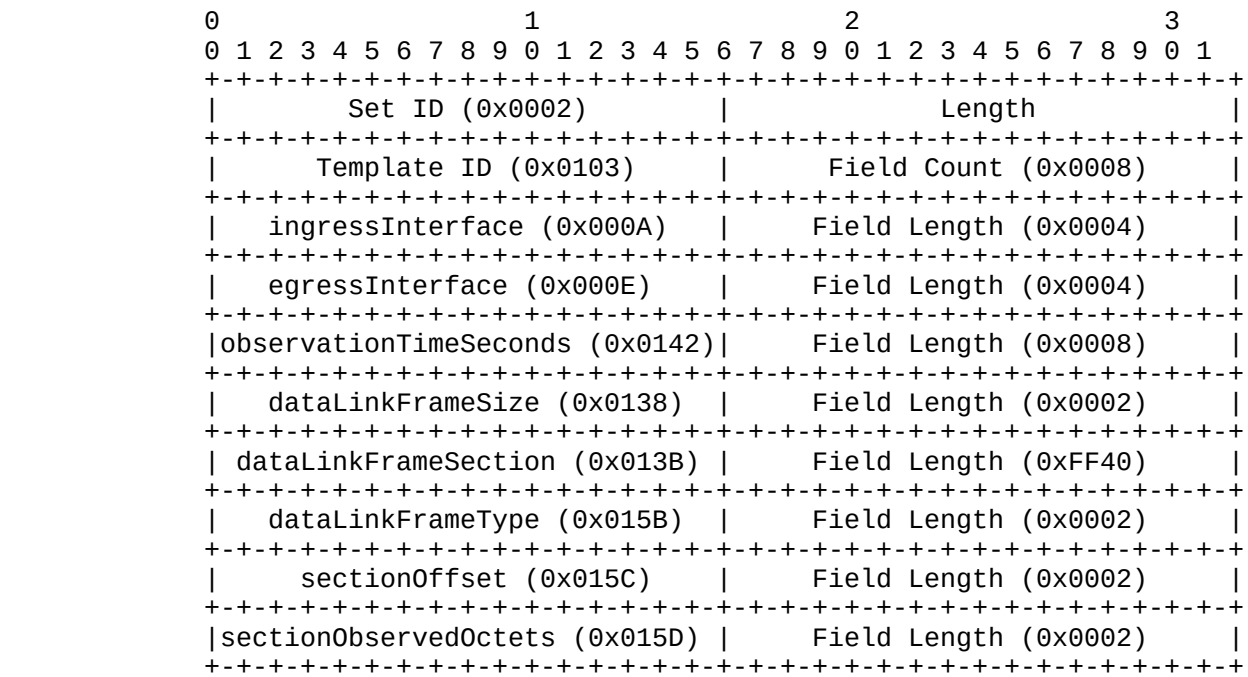


Figure D-1: Template Format Example

Authors' Addresses

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

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

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

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

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

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

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

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

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

Abstract

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

Status of this Memo

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

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

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

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

Copyright Notice

Table of Contents

1. Introduction

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

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

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

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

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

2. Terminology

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

3. Issues and Recommendations Regarding IPFIX over DTLS/UDP

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

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

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

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

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

3.1. Undetected Collector Crashes

3.1.1. Problem Description

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

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

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

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

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

3.1.2. Recommendation

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

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

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

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

3.1.3. Alternative Workarounds

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

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

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

   To make sure that the Exporter learns quickly about a crashed

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

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

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

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

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

3.2. Incorrect Path MTU Values

3.2.1. Problem Description

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

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

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

3.2.2. Recommendation

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

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

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

4. Issues and Recommendations Regarding IPFIX over DTLS/SCTP

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

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

4.1. SCTP-AUTH

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

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

4.2. Renegotiation for DTLS and SCTP-AUTH

4.2.1. Problem Description

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

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

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

4.2.2. Recommendation

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

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

5. Mutual Authentication via Pre-Shared Keys

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

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

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

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

6. Security Considerations

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

Appendix A. Acknowledgements

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

7. References

7.1. Normative References

[RFC4347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security", RFC 4347, April 2006.

[RFC4279] Eronen, P. and H. Tschofenig, "Pre-Shared Key Ciphersuites for Transport Layer Security (TLS)", RFC 4279, December 2005.

[RFC5101] Claise, B., "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information", RFC 5101, January 2008.

7.2. Informative References

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

[RFC1191] Mogul, J. and S. Deering, "Path MTU discovery", RFC 1191, November 1990.

[RFC1981] McCann, J., Deering, S., and J. Mogul, "Path MTU Discovery for IP version 6", RFC 1981, August 1996.

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

[RFC4895] Tuexen, M., Stewart, R., Lei, P., and E. Rescorla, "Authenticated Chunks for the Stream Control Transmission Protocol (SCTP)", RFC 4895, August 2007.

[RFC4960] Stewart, R., "Stream Control Transmission Protocol", RFC 4960, September 2007.

[RFC5153] Boschi, E., Mark, L., Quittek, J., Stiemerling, M., and P. Aitken, "IP Flow Information Export (IPFIX) Implementation Guidelines", RFC 5153, April 2008.

[RFC6083] Tuexen, M., Seggelmann, R., and E. Rescorla, "Datagram Transport Layer Security (DTLS) for Stream Control Transmission Protocol (SCTP)", RFC 6083, January 2011.

[VERMONT] "VERMONT (VERsatile MONitoring Toolkit)", Homepage http://vermont.berlios.de/, 2010.

[OpenSSL] "OpenSSL Cryptography and SSL/TLS Toolkit", Homepage http://www.openssl.org/, 2010.

[I-D.seggelmann-tls-dtls-heartbeat] Seggelmann, R., Tuexen, M., and M. Williams, "Transport Layer Security and Datagram Transport Layer Security Heartbeat Extension", draft-seggelmann-tls-dtls-heartbeat-02 (work in progress), February 2010.

[I-D.ietf-tsvwg-sctpsocket] Stewart, R., Tuexen, M., Poon, K., Lei, P., and V. Yasevich, "Sockets API Extensions for Stream Control Transmission Protocol (SCTP)", draft-ietf-tsvwg-sctpsocket-27 (work in progress), March 2011.

Authors' Addresses

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

Email: mentz@in.tum.de

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

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

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

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

IPFIX Working Group
Internet-Draft
Intended status: BCP
Expires: September 10, 2011

B. Trammell
ETH Zurich
B. Claise
Cisco Systems, Inc.

March 9, 2011

Guidelines for Authors and Reviewers of IPFIX Information Elements
draft-trammell-ipfix-ie-doctors-01.txt

Abstract

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

Status of this Memo

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

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

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

This Internet-Draft will expire on September 10, 2011.

Table of Contents

1. Introduction

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

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

1.1. Intended Audience and Usage

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

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

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

1.2. Overview of relevant IPFIX documents

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

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

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

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

2. Terminology

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

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

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. How to apply IPFIX

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

- The application's data flow is fundamentally unidirectional. IPFIX is a "push" protocol, supporting only the export of information from a sender (an Exporting Process) to a receiver (a Collecting Process). Request-response interactions are not supported by IPFIX.

- The application handles discrete event information, or information to be periodically reported. IPFIX is particularly well suited to representing events, which can be scoped in time.

- The application handles information about network entities. IPFIX's information model is network-oriented, so network management applications have many opportunities for information model reuse.

- The application requires a small number of arrangements of data structures relative to the number of records it handles. The template-driven self-description mechanism used by IPFIX excels at handling large volumes of identically structured data, compared to representations which define structure inline with data (such as XML).

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

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

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

4. Defining new Information Elements

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

- The Information Element MUST be sufficiently unique within the registry. A proposed Information Elements which is a substantial duplicate of an exiting Information Element is to be represented using the existing Element.

- The Information Element SHOULD contain minimal internal structure; complex information should be represented with multiple simple Information Elements to be exported in parallel, as in Section 4.5.

- The Information Element SHOULD be generally applicable to the application at hand, which SHOULD be of general interest to the community. Information Elements representing information about proprietary or nonstandard applications SHOULD be represented using enterprise-specific Information Elements as detailed in section 6.2 of [RFC5101].

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

4.1. Information Element naming

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

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

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

4.2. Information Element data types

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

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

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

4.3. Information Element numbering

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

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

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

4.4. Ancillary Information Element properties

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

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

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

4.5. Internal structure in Information Elements

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

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

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

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

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

4.6. Enumerated Values and Subregistries

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

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

4.7. Reversibility as per RFC 5103

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

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

5. The Information Element Lifecycle: Revision and Deprecation

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

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

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

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

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

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

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

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

- the Information Element definition has an error or shortcoming which cannot be permissibly changed as above; or
- the deprecation harmonizes with an external reference which was itself deprecated through that reference's accepted deprecation method; or
- changes in the IPFIX Protocol or its extensions, or in community understanding thereof, allow the information represented by the Information Element to be represented in a more efficient or convenient way. Deprecation in this circumstance additionally

requires the assent of the IPFIX Working Group, and should be specified in the Internet Draft(s) defining the protocol change.

A request for deprecation is sent to IANA, which passes it to the appointed experts and a responsible Operations Area Director for review; if there is no objection to the change from any appointed expert, IANA makes the change in the Information Element Registry according to its internal procedures. When deprecating an Information Element, the Information Element description MUST be updated to explain the deprecation, as well as to refer to any new Information Elements created to replace the deprecated Information Element.

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

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

6. When not to define new Information Elements

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

6.1. Maximizing reuse of existing Information Elements

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

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

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

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

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

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

6.2. Applying enterprise-specific Information Elements

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

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

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

While work within the IETF generally does not fall into these categories, enterprise-specific Information Elements are also useful for pre-standardization testing of a new IPFIX application. While performing initial development and interoperability testing of a new application, the Information Elements used by the application SHOULD NOT be submitted to IANA for inclusion in the registry. Instead, these experimental Information Elements SHOULD be represented as enterprise-specific until their definitions are finalized, then transitioned from enterprise-specific to IANA-defined upon finalization.

7. Applying IPFIX to non-Flow Applications

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

Non-flow information defining associations or key-value pairs, on the

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

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

8. Writing Internet-Drafts for IPFIX Applications

When a new application is complex enough to require additional clarification or specification as to the use of the defined Information Elements, this may be given in an Internet-Draft. Internet-Drafts for new IPFIX applications are best submitted to a Working Group with expertise in the area of the new application, or as independent submissions.

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

8.1. Example Information Element Definition

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

Description: Description goes here.

Data Type: Data type goes here; obligatory

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

Units: Units, if any, go here; optional

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

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

ElementId: TBD1

8.2. Defining Recommended Templates

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

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

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

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

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

9. A Textual Format for Specifying Information Elements and Templates

The examples given above are all expressed using bitmap diagrams of the respective Templates. These are illustrative of the wire representation of simple Templates, but not particularly readable for more complicated recommended Templates, provide no support for rapid implementation of new Templates, and do not adequately convey the optional nature of ordering and additional Information Elements as above. Therefore, we define a RECOMMENDED textual format for specifying Information Elements and Templates in Internet-Drafts in this section.

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

9.1. Information Element Specifiers

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

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

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

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

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

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

name(number)<type>[size]

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

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

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

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

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

Example fully-qualified IESpecs follow:

octetDeltaCount(1)<unsigned64>[8]

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

sourceIPv4Address(8)<ipv4Address>

wlanSSID(146)<string>[v]

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

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

Example partial IESpecs follow:

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

9.2. Specifying Templates

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

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

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

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

9.3. Specifying IPFIX Structured Data

IESpecs can also be used to illustrate the structure of the information exported using the IPFIX Structured Data extension [I-D.ietf-ipfix-structured-data]. Here, the semantics of the structured data elements are specified using contexts, and the information elements within each structured data element follow the structured data element, prefixed with + to show they are contained therein. Arbitrary nesting of structured data elements is possible by using multiple + signs in the prefix. For example, a basic list of IP addresses with "one or more" semantics would be expressed using parially qualified IESpecs as follows:

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

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

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

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

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

10. Security Considerations

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

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

These security considerations MAY also be stressed in an accompanying Internet-Draft, as in Section 8. For example, the "Packet Sampling (PSAMP) Protocols Specification" [RFC5476] specifies: "In the basic Packet Report, a PSAMP Device exports some number of contiguous bytes from the start of the packet, including the packet header (which includes link layer, network layer and other encapsulation headers) and some subsequent bytes of the packet payload. The PSAMP Device SHOULD NOT export the full payload of conversations, as this would mean wiretapping [RFC2804]. The PSAMP Device MUST respect local privacy laws."

11. IANA Considerations

This document defines a set of guidelines for IANA with respect to the management of the IPFIX Information Element registry and associated subregistries located at http://www.iana.org/assignments/ipfix [iana-ipfix-assignments].

Specifically:

[TODO: collect considerations from the rest of the document and put them here.]

12. Acknowledgements

[TODO]

13. Open Issues

- TBD: determine how or whether to handle IE versioning; this should be discussed in WG.
- TBD: refer to the Draft Standards version of RFC5101 and RFC5102, once these are written.
- TBD: Do we want the IE-Doctors to be a formal directorate under the OPS area? What can we take from the experience of PMOL? This should be handled in discussions with IANA, the IESG, and the IPFIX WG.
- add examples everywhere (including sipclf)

14. References

14.1. Normative References

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

[RFC5101] Claise, B., "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information", RFC 5101, January 2008.

[RFC5102] Quittek, J., Bryant, S., Claise, B., Aitken, P., and J. Meyer, "Information Model for IP Flow Information Export", RFC 5102, January 2008.

[RFC5103] Trammell, B. and E. Boschi, "Bidirectional Flow Export Using IP Flow Information Export (IPFIX)", RFC 5103, January 2008.

[RFC5610] Boschi, E., Trammell, B., Mark, L., and T. Zseby, "Exporting Type Information for IP Flow Information Export (IPFIX) Information Elements", RFC 5610, July 2009.

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

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an

IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

14.2. Informative References

[RFC2804] IAB and IESG, "IETF Policy on Wiretapping", RFC 2804, May 2000.

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

[RFC4181] Heard, C., "Guidelines for Authors and Reviewers of MIB Documents", BCP 111, RFC 4181, September 2005.

[RFC5153] Boschi, E., Mark, L., Quittek, J., Stiemerling, M., and P. Aitken, "IP Flow Information Export (IPFIX) Implementation Guidelines", RFC 5153, April 2008.

[RFC5470] Sadasivan, G., Brownlee, N., Claise, B., and J. Quittek, "Architecture for IP Flow Information Export", RFC 5470, March 2009.

[RFC5471] Schmoll, C., Aitken, P., and B. Claise, "Guidelines for IP Flow Information Export (IPFIX) Testing", RFC 5471, March 2009.

[RFC5472] Zseby, T., Boschi, E., Brownlee, N., and B. Claise, "IP Flow Information Export (IPFIX) Applicability", RFC 5472, March 2009.

[RFC5473] Boschi, E., Mark, L., and B. Claise, "Reducing Redundancy in IP Flow Information Export (IPFIX) and Packet Sampling (PSAMP) Reports", RFC 5473, March 2009.

[RFC5474] Duffield, N., Chiou, D., Claise, B., Greenberg, A., Grossglauser, M., and J. Rexford, "A Framework for Packet Selection and Reporting", RFC 5474, March 2009.

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

[RFC5477] Dietz, T., Claise, B., Aitken, P., Dressler, F., and G. Carle, "Information Model for Packet Sampling Exports", RFC 5477, March 2009.

[RFC5655] Trammell, B., Boschi, E., Mark, L., Zseby, T., and A. Wagner, "Specification of the IP Flow Information Export

(IPFIX) File Format", RFC 5655, October 2009.

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

[I-D.ietf-ipfix-anon]
Boschi, E. and B. Trammell, "IP Flow Anonymization Support", draft-ietf-ipfix-anon-06 (work in progress), January 2011.

[iana-ipfix-assignments]
Internet Assigned Numbers Authority, "IP Flow Information Export Information Elements (http://www.iana.org/assignments/ipfix/ipfix.xml)".

Authors' Addresses

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

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

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

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