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

B. Trammell
E. Boschi
ETH Zurich
A. Wagner
Consecom AG

October 25, 2010

Exporting Aggregated Flow Data using the IP Flow Information Export (IPFIX) Protocol
draft-trammell-ipfix-a9n-01.txt

Abstract

This document describes the export of aggregated Flow information using IPFIX. An Aggregated Flow is essentially an IPFIX Flow representing packets from zero or more original Flows, within an externally imposed time interval. The document describes Aggregated Flow export within the framework of IPFIX Mediators and defines an interoperable, implementation-independent method for Aggregated Flow export.

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.

Copyright Notice

Table of Contents

1. Introduction

The aggregation of packet data into flows serves a variety of different purposes, as noted in the requirements [RFC3917] and applicability statement [RFC5472] for the IP Flow Information Export (IPFIX) protocol [RFC5101]. Aggregation beyond the flow level, into records representing multiple Flows, is a common analysis and data reduction technique as well, with applicability to large-scale network data analysis, archiving, and inter-organization exchange. This applicability in large-scale situations, in particular, led to the inclusion of aggregation as part of the IPFIX Mediators Problem Statement [RFC5982], and the definition of an Intermediate Aggregation Process in the Mediator framework [I-D.ietf-ipfix-mediators-framework].

The Mediator framework offered an initial but inexhaustive treatment of the topic of aggregation. This document expands on the definitions presented there, providing an implementation-neutral, interoperable specification of an Intermediate Aggregation Process which can operate within the Mediator framework or independent thereof.

Aggregation is part of a wide variety of applications, including traffic matrix calculation, generation of time series data for visualizations or anomaly detection, and data reduction. Depending on the keys used for aggregation, it may have an anonymising affect on the data. Aggregation can take place at one of any number of locations within a measurement infrastructure. Exporters may export aggregated Flow information simply as normal flow information, by performing aggregation after metering but before export. IPFIX Mediators are particularly well suited to performing aggregation, as they can collect information from multiple original exporters at geographically and topologically distinct observation points.

Aggregation as defined and described in this document covers a superset of the applications defined in [RFC5982], including 5.1 "Adjusting Flow Granularity (herein referred to as Key Aggregation), 5.4 "Time Composition" (herein referred to as Interval Combination), and 5.5 "Spatial Composition".

Note that an Intermediate Aggregation process may be applied to data collected from multiple Observation Points, as aggregation is natural to apply for data reduction when concentrating measurement data. This document specifically does not address the architectural and protocol issues that arise when combining IPFIX data from multiple Observation Points and exporting from a single Mediator, as these issues are general to Mediation in general. These are treated in detail in the Mediator Protocol [I-D.claise-ipfix-mediation-protocol]

document.

Since aggregated flows as defined in the following section are essentially Flows, IPFIX can be used to export [RFC5101] and store [RFC5655] aggregated data "as-is"; there are no changes necessary to the protocol. However, this document further provides a common basis for the application of IPFIX to the handling of aggregated data, through a detailed terminology, model of aggregation operations, methods for original Flow counting and counter distribution across time intervals, and an aggregation metadata representation based upon IPFIX Options.

1.1. Rationale and Scope

This specification of Aggregated Flow export has interoperability and implementation-independence as its two key goals. First, export of Aggregated Flows using the techniques described in this document will result in Flow data which can be collected by Collecting Processes and read by File Readers which do not provide any special support for Aggregated Flow export. An Aggregated Flow is simply a Flow with some additional conditions as to how it is derived.

Second, in Section 5, we specify aggregation in an implementation-independent way. While we must describe the aggregation process in terms of operations due to the interdependencies among them, these operations like the stages in the IPFIX Architecture [RFC5470] are meant to be descriptive as opposed to proscriptive. We specify the flow aggregation process as an intermediate process within the IPFIX Mediator framework [I-D.ietf-ipfix-mediators-framework], and specify a variety of different architectural arrangements for flow aggregation. When exporting aggregation-relevant metadata, we seek to define properties of the set of exported Aggregated Flows, as opposed to the properties of the specific algorithms used to aggregate these Flows. Specifically out of scope for this effort are any definition of a language for defining aggregation operations, or the configuration parameters of Aggregation Processes, as these are necessarily implementation dependent.

From the definition of presented below in Section 2, an Aggregated Flow is a Flow as in [RFC5101], with additional conditions as to the packets making up the Flow. Practically speaking, Aggregated Flows are derived from original Flows, as opposed to a raw packet stream. Key to this definition of Aggregated Flow is how timing affects the process of aggregation, as for the most part flow aggregation takes place within some set of time intervals, which are usually regular and externally imposed, or derived from the flows themselves. Aggregation operations concerning keys, which are often called "spatial aggregation" in the literature, will necessarily impact and

be impacted by these time intervals; aggregation operations concerning these time intervals are often called "temporal aggregation" in the literature. Prior definitions of aggregation attempt to treat temporal and spatial aggregation separately; this document recognizes that this is not possible due to the interdependencies between flows and their time intervals, and defines these operations as interdependent.

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 [RFC2119].

In addition, this document defines the following terms

Aggregated Flow: A Flow, as defined by [RFC5101], derived from a set of zero or more original Flows within a defined time interval. The two primary differences between a Flow and an Aggregated Flow are (1) that the time interval of a Flow is generally derived from information about the timing of the packets comprising the Flow, while the time interval of an Aggregated Flow are generally externally imposed; and (2) that an Aggregated Flow may represent zero packets (i.e., an assertion that no packets were seen for a given Flow Key in a given time interval).

(Intermediate) Aggregation Function: A mapping from a set of zero or more original Flows into a set of Aggregated Flows accross one or more time intervals.

(Intermediate) Aggregation Process: An Intermediate Process, as in [I-D.ietf-ipfix-mediators-framework], hosting an Intermediate Aggregation Function. Note that this definition, together with that given above, updates the definition given in [I-D.ietf-ipfix-mediators-framework] to account for the more precise definition of Aggregated Flow given herein. An Aggregation Process need not be intermediate; that is, while Aggregation Processes will often be deployed within a Mediator, this is not necessarily the case.

Aggregation Interval: A time interval imposed upon an Aggregated Flow. Aggregation Functions may use a regular Aggregation Interval (e.g. "every five minutes", "every calendar month"), though regularity is not necessary. Aggregation intervals may also be derived from the time intervals of the flows being aggregated.

original Flow: A Flow given as input to an Aggregation Function in order to generate Aggregated Flows.

contributing Flow: An original Flow that is partially or completely represented within an Aggregated Flow. Each aggregated Flow is made up of zero or more contributing Flows, and an original flow may contribute to zero or more Aggregated Flows.

3. Use Cases for IPFIX Aggregation

Aggregation, as a common data analysis method, has many applications. When used with a regular Aggregation Interval, it generates time series data from a collection of flows with discrete intervals. Time series data is itself useful for a wide variety of analysis tasks, such as generating parameters for network anomaly detection systems, or driving visualizations of volume per time for traffic with specific characteristics. Traffic matrix calculation from flow data is inherently an aggregation action, by aggregating the flow key down to interface, address prefix, or autonomous system.

Irregular or data-dependent Aggregation Intervals and Key Aggregation operations can be also be used to provide adaptive aggregation of network flow data, providing a lower-resolution view (i.e. more aggregation) on data deemed "less interesting" to a given application, while allowing higher resolution (i.e. less or no aggregation) for data of interest. For example, in a Mediator equipped with traffic classification capabilities for security purposes, potentially malicious flows could be exported directly, while known-good or probably-good flows (e.g. normal web browsing) could be exported simply as time series volumes per web server.

Note that an aggregation operation which removes potentially sensitive information as identified in [I-D.ietf-ipfix-anon] may tend to have an anonymising effect on the Aggregated Flows, as well; however, any application of aggregation as part of a data protection scheme should ensure that all the issues raised in Section 4 of [I-D.ietf-ipfix-anon] are addressed.

4. Aggregation of IP Flows

As stated in Section 2, an Aggregated Flow is simply an IPFIX Flow generated from original Flows by an Aggregation Function. Here, we discuss temporal and spatial aspects of aggregation, present a general model for aggregation, and elaborate and provide examples of specific aggregation operations that may be performed by the Aggregation Process; we use this to define the export of Aggregated Flows in Section 6

4.1. A note on temporal and spatial aggregation

In general, aggregation of data records bearing time information can take place in time (by grouping the original records by time) or in space (by grouping the original records by some other dimension; in the case of IP Flows, this would generally be a flow key.

Temporal aggregation is treated in [I-D.ietf-ipfix-mediators-framework] in section 5.3.2.3, as "[m]erging a set of Data Records within a certain time period into one Flow Record by summing up the counters where appropriate," as well as in the definition of "temporal composition, wherein "multiple consecutive Flow Records with identical Flow Key values are merged into a single Flow Record of longer Flow duration if they arrive within a certain time interval."

Spatial aggregation is treated in [I-D.ietf-ipfix-mediators-framework] in section 5.3.2.3, as "spatial composition", wherein "Data Records sharing common properties are merged into one Flow Record within a certain time period." Even this definition hints at the problem in attempting to treat temporal and spatial aggregation of IP flow data orthogonally.

The issue arises because an IP Flow, as defined in [RFC5101], has three types of properties: flow keys, which "define" the properties common to all packets in the Flow; flow values or non-key fields, which describe the Flow itself; and the time interval of the Flow. The keys and time interval serve to uniquely identify the Flow. When spatially aggregating Flows, these Flows bring their time intervals along with them. The time intervals of the spatially aggregated Flows must either be combined through union, or externally imposed by splitting the original Flow across one or more

To address this subtle interdependency, it is more useful to view an Aggregation Function in terms of the temporal operations of the function, called "interval distribution" herein; and the spatial operations of the function, called "key aggregation" herein; this follows in the general model presented in the following subsection.

4.2. A general operational model for IP Flow aggregation

An Intermediate Aggregation Process consumes original Flows and exports Aggregated Flows, as defined in Section 2. While this document does not define an implementation of an Intermediate Aggregation Process further than this, or the Aggregation Functions that it applies, it can be helpful to partially decompose this function into a set of common operations, in order to more fully examine the effects these operations have.

Aggregation is composed of three general types of operations on original Flows: those that externally impose a time interval, called here the Aggregation Interval; those that derive a new Flow Key for the Aggregated Flows from the original Flow information; and those that aggregate and distribute the resulting non-Flow Key fields accordingly. Most aggregation functions will perform each of these types of operations.

Interval distribution is the external imposition of a time interval onto an original Flow. Note that this may lead to an original Flow contributing to multiple aggregated Flows, if the original Flow's time interval crosses at least one boundary between Aggregation Intervals. Interval Distribution is described in more detail in Section 4.3.

Key aggregation, the derivation of Flow Keys for Aggregated Flows from original Flow information, is made up of two operations: reduction and replacement. Reduction removes Information Elements from the original Flow Key, or otherwise constrains the space of values in the Flow Key (e.g., by replacing IP addresses with /24 CIDR blocks). In replacement, Information Elements derived from fields in the original Flow itself may be added to the Flow Key. Both of these modifications may result in multiple original Flows contributing to the same Aggregated Flow. Key Aggregation is described in more detail in Section 4.4.

Interval distribution and key aggregation together may generate multiple intermediate aggregated Flows covering the same time interval with the same Flow Key; these intermediate Flows must therefore be combined into Aggregated Flows. Non-key values are first distributed among the Aggregated Flows to which an original Flow contributes according to some distribution algorithm (see Section 4.5), and values from multiple contributing Flows are combined using the same operation by which values are combined from packets to form Flows for each Information Element: in general, counters are added, averages are averaged, flags are unioned, and so on. Key aggregation may also introduce new non-key fields, e.g. per-flow average counters, or distinct counters for key fields reduced

out of the Aggregated Flow.

As a result of this final combination and distribution,an Aggregation Function produces at most one Aggregated Flow resulting from a set of original Flows for a given Aggregated Flow Key and Aggregation Interval.

This general model is illustrated in the figure below. Note that within an implementation, these steps may occur in any order, and indeed be combined together in any way.

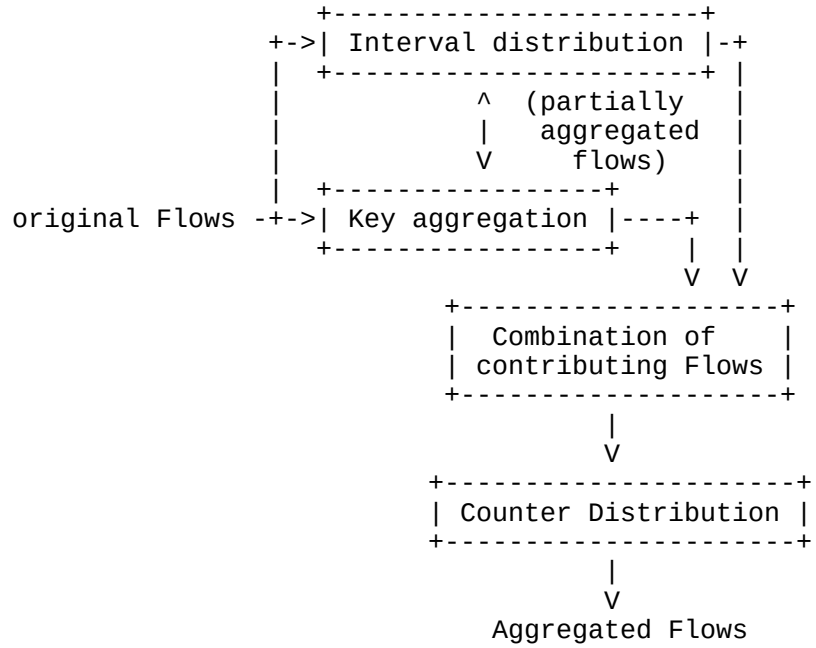

Figure 1: Conceptual model of aggregation operations

4.3. Interval Distribution

Interval Distribution imposes a time interval on the resulting Aggregated Flows. The selection of an interval is a matter for the specific aggregation application. Intervals may be derived from the flows themselves (e.g, an interval may be selected to cover the entire interval containing the set of all flows sharing a given Key) or externally imposed; in the latter case the externally imposed interval may be regular (e.g., every five minutes) or irregular (e.g., to allow for different time resolutions at different times of day, under different network conditions, or indeed for different sets

of original Flows).

The length of the imposed interval itself has tradeoffs. and has tradeoffs. Shorter intervals allow higher resolution aggregated data and, in streaming applications, faster reaction time. Longer intervals lead to greater data reduction and simplified counter distribution. Specifically, counter distribution is greatly simplified by the choice of an interval longer than the duration of longest original Flow, itself generally determined by the original Flow's Metering Process active timeout; in this case an original Flow can contribute to at most two Aggregated Flows, and the more complex value distribution methods become inapplicable.

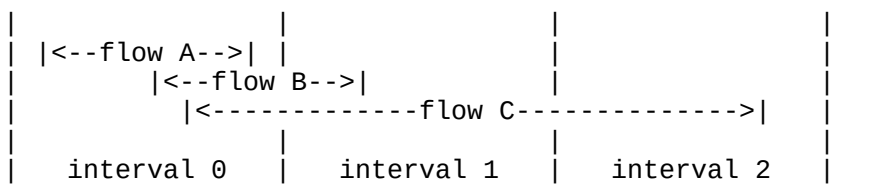

Figure 2: Illustration of interval distribution

In Figure 2, we illustrate three common possibilities for interval distribution as applies with regular intervals to a set of three original Flows. For flow A, the start and end times lie within the boundaries of a single interval 0; therefore, flow A contributes to only one Aggregated Flow. Flow B, by contrast, has the same duration but crosses the boundary between intervals 0 and 1; therefore, it will contribute to two Aggregated Flows, and its counters must be distributed among these flows, though in the two-interval case this can be simplified somewhat simply by picking one of the two intervals, or proportionally distributing between them. Only flows like flow A and flow B will be produced when the interval is chosen to be longer than the duration of longest original Flow, as above. More complicated is the case of flow C, which contributes to more than two flows, and must have its counters distributed according to some policy as in Section 4.5.

4.4. Key Aggregation

Key Aggregation generates a new Flow Key for the Aggregated Flows from the original Flow Keys, non-Key fields in the original Flows, or from correlation of the original Flow information with some external source. There are two basic operations here. First, Aggregated Flow Keys may be derived directly from original Flow Keys through reduction, or the dropping of fields or precision in the original Flow Keys. Second, an Aggregated Flow Key may be derived through replacement, e.g. by removing one or more fields from the original

Flow and replacing them with a fields derived from the removed fields. Replacement may refer to external information (e.g., IP to AS number mappings). Replacement need not replace only key fields; for example, an application aggregating byte counts per flow size in packets would promote the packet count to a Flow Key field.

Key aggregation may also result in the addition of new non-Key fields to the Aggregated Flows, namely original Flow counters and unique reduced key counters; these are treated in more detail in Section 4.6 and Section 4.7, respectively.

In any Key Aggregation operation, reduction and/or replacement may be applied any number of times in any order. Which of these operations are supported by a given implementation is implementation- and application-dependent. Key Aggregation may aggregate original Flows with different sets of Flow Key fields; only the Flow Keys of the resulting Aggregated Flows of any given Key Aggregation operation need contain the same set of fields.

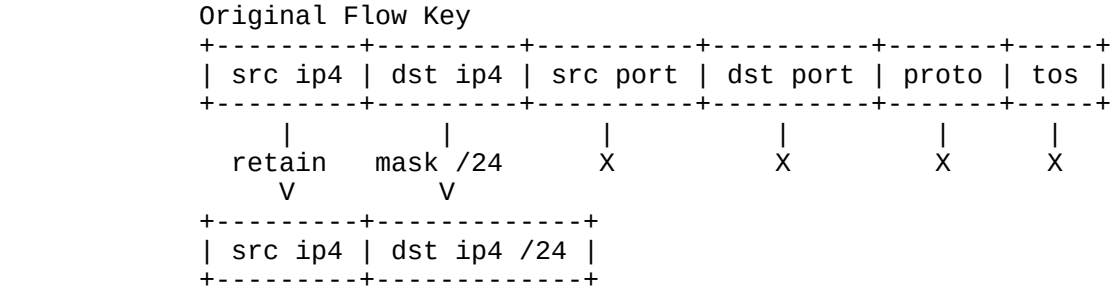

Figure 3: Illustration of key aggregation by reduction

Figure 3 illustrates an example reduction operation, aggregation by source address and destination class C network. Here, the port, protocol, and type-of-service information is removed from the flow key, the source address is retained, and the destination address is masked by dropping the low 8 bits.

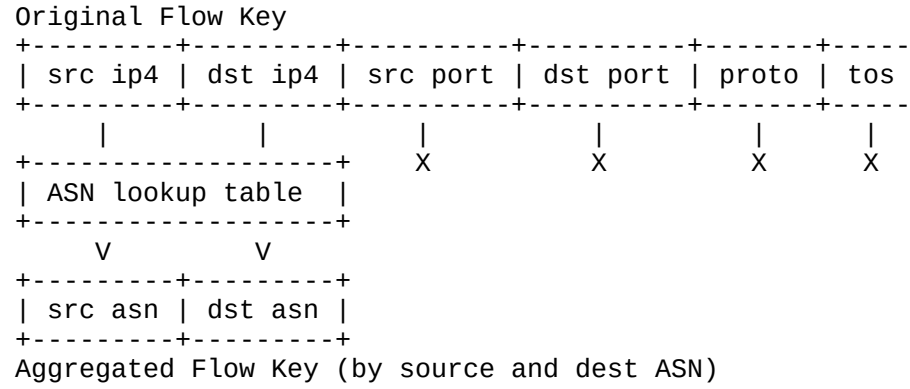

Figure 4: Illustration of key aggregation by reduction and replacement

Figure 4 illustrates an example reduction and replacement operation, aggregation by source and destination ASN without ASN information available in the original Flow. Here, the port, protocol, and type-of-service information is removed from the flow key, while the source and destination addresses are run though an IP address to ASN lookup table, and the Aggregated Flow key is made up of the resulting source and destination ASNs.

4.5. Aggregating and Distributing Counters

In general, counters in Aggregated Flows are treated the same as in any Flow. Each counter is independently is calculated as if it were derived from the set of packets in the original flow. For the most part, when aggregating original Flows into Aggregated Flows, this is simply done by summation.

When the Aggregation Interval is guaranteed to be longer than the longest original Flow, a Flow can cross at most one Interval boundary, and will therefore contribute to at most two Aggregated Flows. Most common in this case is to arbitrarily but consistently choose to account the original Flow’s counters either to the first or the last aggregated Flow to which it could contribute.

However, this becomes more complicated when the Aggregation Interval is shorter than the longest original Flow in the source data. In such cases, each original Flow can incompletely cover one or more time intervals, and apply to one or more Aggregated Flows; in this case, the Aggregation Process must distribute the counters in the original Flows across the multiple Aggregated Flows. There are several methods for doing this, listed here in roughly increasing order of complexity and accuracy.

End Interval: The counters for an original Flow are added to the counters of the appropriate Aggregated Flow containing the end time of the original Flow.

Start Interval: The counters for an original Flow are added to the counters of the appropriate Aggregated Flow containing the start time of the original Flow.

Mid Interval: The counters for an original Flow are added to the counters of a single appropriate Aggregated Flow containing some timestamp between start and end time of the original Flow.

Simple Uniform Distribution: Each counter for an original Flow is divided by the number of time intervals the original Flow covers (i.e., of appropriate Aggregated Flows sharing the same Flow Key), and this number is added to each corresponding counter in each Aggregated Flow.

Proportional Uniform Distribution: Each counter for an original Flow is divided by the number of time _units_ the original Flow covers, to derive a mean count rate. This mean count rate is then multiplied by the number of time units in the intersection of the duration of the original Flow and the time interval of each Aggregated Flow. This is like simple uniform distribution, but accounts for the fractional portions of a time interval covered by an original Flow in the first and last time interval.

Simulated Process: Each counter of the original Flow is distributed among the intervals of the Aggregated Flows according to some function the Aggregation Process uses based upon properties of Flows presumed to be like the original Flow. For example, bulk transfer flows might follow a more or less proportional uniform distribution, while interactive processes are far more bursty.

Direct: The Aggregation Process has access to the original packet timings from the packets making up the original Flow, and uses these to distribute or recalculate the counters.

A method for exporting the distribution of counters across multiple Aggregated Flows is detailed in Section 6.3. In any case, counters MUST be distributed across the multiple Aggregated Flows in such a way that the total count is preserved, within the limits of accuracy of the implementation (e.g., inaccuracy introduced by the use of floating-point numbers is tolerable). This property allows data to be aggregated and re-aggregated without any loss of original count information. To avoid confusion in interpretation of the aggregated data, all the counters for a set of given original Flows SHOULD be distributed via the same method.

4.6. Counting Original Flows

When aggregating multiple original Flows into an Aggregated Flow, it is often useful to know how many original Flows are present in the Aggregated Flow. This document introduces four new information elements in Section 6.2 to export these counters.

There are two possible ways to count original Flows, which we call here conservative and non-conservative. Conservative flow counting has the property that each original Flow contributes exactly one to the total flow count within a set of aggregated Flows. In other words, conservative flow counters are distributed just as any other counter, except each original Flow is assumed to have a flow count of one. When a count for an original Flow must be distributed across a set of Aggregated Flows, and a distribution method is used which does not account for that original Flow completely within a single Aggregated Flow, conservative flow counting requires a fractional representation.

By contrast, non-conservative flow counting is used to count how many flows are represented in an Aggregated Flow. Flow counters are not distributed in this case. An original Flow which is present within N Aggregated Flows would add N to the sum of non-conservative flow counts, one to each Aggregated Flow. In other words, the sum of conservative flow counts over a set of Aggregated Flows is always equal to the number of original Flows, while the sum of non-conservative flow counts is strictly greater than or equal to the number of original Flows.

For example, consider flows A, B, and C as illustrated in Figure 2. Assume that the key aggregation step aggregates the keys of these three flows to the same aggregated flow key, and that start interval counter distribution is in effect. The conservative flow count for interval 0 is 3 (since flows A, B, and C all begin in this interval), and for the other two intervals is 0. The non-conservative flow count for interval 0 is also 3 (due to the presence of flows A, B, and C), for interval 1 is 2 (flows B and C), and for interval 2 is 1 (flow 0). The sum of the conservative counts 3 + 0 + 0 = 3, the number of original Flows; while the sum of the non-conservative counts 3 + 2 + 1 = 6.

4.7. Counting Distinct Key Values

One common case in aggregation is counting distinct values that were reduced away during key aggregation. For example, consider an application counting destinations contacted per host, a common case in host characterization or anomaly detection. Here, the Aggregation Process needs a way to export this distinct key count information.

For such applications, a distinctCountOf(key name) Information Element should be registered with IANA to represent these cases. [EDITOR'S NOTE: There is an open question as to the best way to do this: either through the registration of Information Elements for common cases in this draft, the registration of Information Elements on demand, or the definition of a new Information Element space for distinct counts bound to a PEN, as in [RFC5103].]

4.8. Exact versus Approximate Counting during Aggregation

In certain circumstances, particularly involving aggregation by devices with limited resources, and in situations where exact aggregated counts are less important than relative magnitudes (e.g. driving graphical displays), counter distribution during key aggregation may be performed by approximate counting means (e.g. Bloom filters). The choice to use approximate counting is implementation- and application-dependent.

4.9. Time Composition

Time Composition as in section 5.4 of [RFC5982] (or interval combination) is a special case of aggregation, where interval distribution imposes longer intervals on flows with matching keys and "chained" start and end times, without any key reduction, in order to join long-lived Flows which may have been split (e.g., due to an active timeout shorter than the Flow.) Here, no Key Aggregation is applied, and the Aggregation Interval is chosen on a per-Flow basis to cover the interval spanned by the set of aggregated Flows. This may be applied alone in order to normalize split Flows, or in combination with other aggregation functions in order to obtain more accurate original Flow counts.

5. Aggregation in the IPFIX Architecture

The techniques described in this document can be applied to IPFIX data at three stages within the collection infrastructure: on initial export, within a mediator, or after collection, as shown in Figure 5.

[EDITOR'S NOTE: determine where this lives: in the introduction or down here? Note explicitly that an IAP may live outside a mediator. Check both these figures for parallels to mediator framework.]

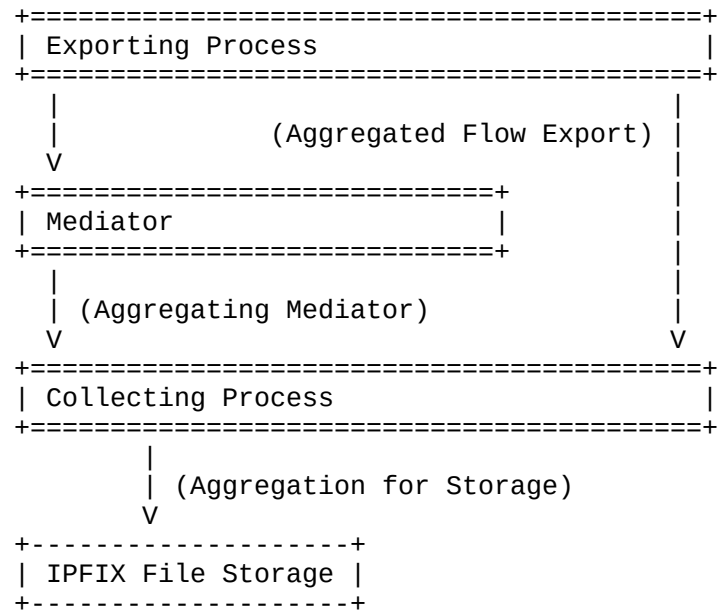

Figure 5: Potential Aggregation Locations

Aggregation can be applied for either intermediate or final analytic purposes. In certain circumstances, it may make sense to export Aggregated Flows from an Exporting Process, for example, if the Exporting Process is designed to drive a time-series visualization directly. Note that this case, where the Aggregation Process is essentially integrated into the Metering Process, is essentially covered by the IPFIX architecture [RFC5470]: the flow keys used are simply a subset of those that would normally be used. A Metering Process in this arrangement MAY choose to simulate the generation of larger flows in order to generate original flow counts, if the application calls for compatibility with an Aggregation Process deployed in a separate location.

Deployment of an Intermediate Aggregation Process within a Mediator [RFC5982] is a much more flexible arrangement. Here, the Mediator consumes original Flows and produces aggregated Flows; this arrangement is suited to any of the use cases detailed in Section 3. In a mediator, aggregation can be applied as well to aggregating original Flows from multiple sources into a single stream of aggregated Flows; the architectural specifics of this arrangement are not addressed in this document, which is concerned only with the aggregation operation itself; see [I-D.claise-ipfix-mediation-protocol] for details.

In the specific case that an Aggregation Process is employed for data reduction for storage purposes, it can take original Flows from a Collecting Process or File Reader and pass Aggregated Flows to a File Writer for storage.

The data flows into and out of an Intermediate Aggregation Process are showin in Figure 6.

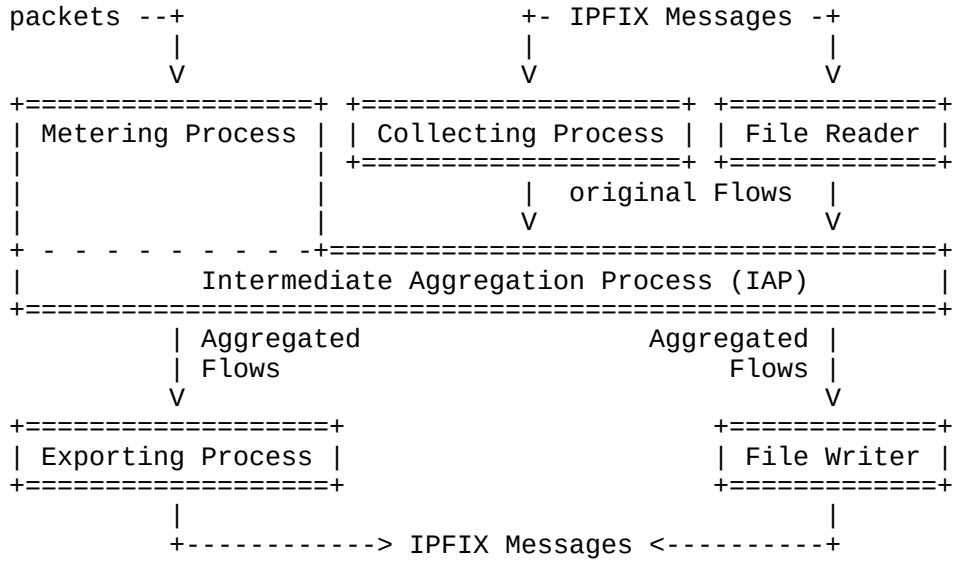

Figure 6: Data flows through the aggregation process

6. Export of Aggregated IP Flows using IPFIX

In general, Aggregated Flows are exported in IPFIX as any normal Flow. However, certain aspects of aggregated flow export benefit from additional guidelines, or new Information Elements to represent aggregation metadata or information generated during aggregation. These are detailed in the following subsections.

6.1. Time Interval Export

Since an Aggregated Flow is simply a Flow, the existing timestamp Information Elements in the IPFIX Information Model (e.g., flowStartMilliseconds, flowEndNanoseconds) are sufficient to specify the time interval for aggregation. Therefore, this document specifies no new aggregation-specific Information Elements for exporting time interval information.

Each Aggregated Flow SHOULD contain both an interval start and interval end timestamp. If an exporter of Aggregated Flows omits the interval end timestamp from each Aggregated Flow, the time interval for Aggregated Flows within an Observation Domain and Transport Session MUST be regular and constant. However, note that this approach might lead to interoperability problems when exporting Aggregated Flows to non-aggregation-aware Collecting Processes and downstream analysis tasks; therefore, an Exporting Process capable of exporting only interval start timestamps MUST provide a configuration option to export interval end timestamps as well.

6.2. Flow Count Export

The following four Information Elements are defined to count original Flows as discussed in Section 4.6.

6.2.1. originalFlowsPresent Information Element

Description: The non-conservative count of original Flows contributing to this Aggregated Flow. Non-conservative counts need not sum to the original count on re-aggregation.

Abstract Data Type: unsigned64

ElementId: TBD1

Status: Proposed

6.2.2. originalFlowsInitiated InformationElement

Description: The conservative count of original Flows whose first packet is represented within this Aggregated Flow. Conservative counts must some to the original count on re-aggregation.

Abstract Data Type: unsigned64

ElementId: TBD2

Status: Proposed

6.2.3. originalFlowsCompleted InformationElement

Description: The conservative count of original Flows whose last packet is represented within this Aggregated Flow. Conservative counts must some to the original count on re-aggregation.

Abstract Data Type: unsigned64

ElementId: TBD3

Status: Proposed

6.2.4. originalFlows InformationElement

Description: The conservative count of original Flows contributing to this Aggregated Flow; may be distributed via any of the methods described in Section 4.5.

Abstract Data Type: float64

ElementId: TBD4

Status: Proposed

6.3. Aggregate Counter Distibution Export

When exporting counters distributed among Aggregated Flows, as described in Section 4.5, the Exporting Process MAY export an Aggregate Counter Distribution Record for each Template describing Aggregated Flow records; this Options Template is described below. It uses the valueDistributionMethod Information Element, also defined below. Since in many cases distribution is simple, accounting the counters from contributing Flows to the first Interval to which they contribute, this is default situation, for which no Aggregate Counter Distribution Record is necessary; Aggregate Counter Distribution Records are only applicable in more exotic situations, such as using an Aggregation Interval smaller than the durations of original Flows.

6.3.1. Aggregate Counter Distribution Options Template

This Options Template defines the Aggregate Counter Distribution Record, which allows the binding of a value distribution method to a Template ID. This is used to signal to the Collecting Process how the counters were distributed. The fields are as below:

IE	Description
templateId [scope]	The Template ID of the Template defining the Aggregated Flows to which this distribution option applies. This Information Element MUST be defined as a Scope Field.

valueDistributionMethod	The method used to distribute the counters for the Aggregated Flows defined by the associated Template.

6.3.2. valueDistributionMethod Information Element

Description: A description of the method used to distribute the counters from contributing Flows into the Aggregated Flow records described by an associated Template. The method is deemed to apply to all the non-key Information Elements in the referenced Template for which value distribution is a valid operation; if the originalFlowsInitiated and/or originalFlowsCompleted Information Elements appear in the Template, they are not subject to this distribution method, as they each infer their own distribution method. The distribution methods are taken from Section 4.5 and encoded as follows:

Value	Description
1	Start Interval: The counters for an original Flow are added to the counters of the appropriate Aggregated Flow containing the start time of the original Flow. This should be assumed the default if value distribution information is not available at a Collecting Process for an Aggregated Flow.
2	End Interval: The counters for an original Flow are added to the counters of the appropriate Aggregated Flow containing the end time of the original Flow.
3	Mid Interval: The counters for an original Flow are added to the counters of a single appropriate Aggregated Flow containing some timestamp between start and end time of the original Flow.
4	Simple Uniform Distribution: Each counter for an original Flow is divided by the number of time intervals the original Flow covers (i.e., of appropriate Aggregated Flows sharing the same Flow Key), and this number is added to each corresponding counter in each Aggregated Flow.

5	Proportional Uniform Distribution: Each counter for an original Flow is divided by the number of time _units_ the original Flow covers, to derive a mean count rate. This mean count rate is then multiplied by the number of time units in the intersection of the duration of the original Flow and the time interval of each Aggregated Flow. This is like simple uniform distribution, but accounts for the fractional portions of a time interval covered by an original Flow in the first and last time interval.
6	Simulated Process: Each counter of the original Flow is distributed among the intervals of the Aggregated Flows according to some function the Aggregation Process uses based upon properties of Flows presumed to be like the original Flow. This is essentially an assertion that the Aggregation Process has no direct packet timing information but is nevertheless not using one of the other simpler distribution methods. The Aggregation Process specifically makes no assertion as to the correctness of the simulation.
7	Direct: The Aggregation Process has access to the original packet timings from the packets making up the original Flow, and uses these to distribute or recalculate the counters.

Abstract Data Type: unsigned8

ElementId: TBD5

Status: Proposed

7. Examples

[TODO]

8. Security Considerations

[TODO]

9. IANA Considerations

[TODO: add all IEs defined in Section 6.]

10. Acknowledgments

Many thanks to Benoit Claise for his thorough review of this work. This work is materially supported by the European Union Seventh Framework Programme under grant agreement 257315 (DEMONS).

11. References

11.1. 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.

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

11.2. Informative References

[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.

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

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

[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.

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

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

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

[I-D.ietf-ipfix-anon] Boschi, E. and B. Trammell, "IP Flow Anonymisation Support", draft-ietf-ipfix-anon-05 (work in progress), October 2010.

[I-D.ietf-ipfix-mediators-framework] Kobayashi, A., Claise, B., Muenz, G., and K. Ishibashi, "IPFIX Mediation: Framework", draft-ietf-ipfix-mediators-framework-08 (work in progress), August 2010.

[I-D.claise-ipfix-mediation-protocol] Claise, B., Kobayashi, A., and B. Trammell, "Specification of the Protocol for IPFIX Mediations", draft-claise-ipfix-mediation-protocol-01 (work in progress), March 2010.

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

Elisa Boschi
Swiss Federal Institute of Technology Zurich
Gloriastrasse 35
8092 Zurich
Switzerland

Email: boschie@tik.ee.ethz.ch

Arno Wagner
Consecom AG
Bellariastrasse 12
8002 Zurich
Switzerland

Email: arno@wagner.name