Network Working Group
Request for Comments: 5470
Category: Informational

G. Sadasivan
Rohati Systems
N. Brownlee
CAIDA | The University of Auckland
B. Claise
Cisco Systems, Inc.
J. Quittek
NEC

March 2009

Architecture for IP Flow Information Export

Status of This Memo

Copyright Notice

Abstract

This memo defines the IP Flow Information eXport (IPFIX) architecture for the selective monitoring of IP Flows, and for the export of measured IP Flow information from an IPFIX Device to a Collector.

Table of Contents

1. Introduction

There are several applications, e.g., usage-based accounting, traffic profiling, traffic engineering, attack/intrusion detection, quality-of-service (QoS) monitoring, that require Flow-based IP traffic measurements. It is therefore important to have a standard way of exporting information related to IP Flows. This document defines an architecture for IP traffic Flow monitoring, measuring, and exporting. It provides a high-level description of an IPFIX Device's key components and their functions.

1.1. Document Scope

This document defines the architecture for IPFIX. Its main objectives are to:

- Describe the key IPFIX architectural components, consisting of (at least) IPFIX Devices and Collectors communicating using the IPFIX protocol.

- Define the IPFIX architectural requirements, e.g., recovery, security, etc.

- Describe the characteristics of the IPFIX protocol.

1.2. IPFIX Documents Overview

The IPFIX protocol provides network administrators with access to IP Flow information. This document specifies the architecture for the export of measured IP Flow information from an IPFIX Exporting Process to a Collecting Process, per the requirements defined in RFC 3917 [1]. The IPFIX protocol document, RFC 5101 [3], specifies how IPFIX data records and templates are carried via a congestion-aware transport protocol, from IPFIX Exporting Process to IPFIX Collecting Process. IPFIX has a formal description of IPFIX information elements (fields), their name, type, and additional semantic information, as specified in RFC 5102 [2]. Finally, RFC 5472 [4] describes what type of applications can use the IPFIX protocol and how they can use the information provided. Furthermore, it shows how the IPFIX framework relates to other architectures and frameworks.

Note that the IPFIX system does not provide for remote configuration of an IPFIX device. Instead, implementors must provide an effective way to configure their IPFIX devices.

2. Terminology

The definitions of basic IPFIX terms such as IP Traffic Flow, Exporting Process, Collecting Process, Observation Point, etc., are semantically identical with those found in the IPFIX requirements document, RFC 3917 [1]. Some of the terms have been expanded for more clarity when defining the protocol. Additional definitions required for the architecture have also been defined. For terms that are defined here and in RFC 5101 [3], the definitions are equivalent in both documents.

* Observation Point

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

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

* Observation Domain

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

* IP Traffic Flow or Flow

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

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

1. one or more packet header fields (e.g., destination IP address), transport header fields (e.g., destination port number), or application header fields (e.g., RTP header fields [5]).

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

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

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

This definition covers the range from a Flow containing all packets observed at a network interface to a Flow consisting of just a single packet between two applications. It includes packets selected by a sampling mechanism.

* Flow Key

Each of the fields that:

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

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

3. is derived from packet treatment (e.g., Autonomous System (AS) number), and

4. is used to define a Flow

is termed a Flow Key.

* Flow Record

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

* Metering Process

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

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

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

* Exporting Process

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

* Exporter

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

* IPFIX Device

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

* Collecting Process

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

* Collector

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

* Template

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

* Control Information, Data Stream

  The information that needs to be exported from the IPFIX Device can be classified into the following categories:

  Control Information

    This includes the Flow definition, selection criteria for packets within the Flow sent by the Exporting Process, and templates describing the data to be exported. Control Information carries all the information needed for the end-points to understand the IPFIX protocol, and specifically for the Collector to understand and interpret the data sent by the sending Exporter.

  Data Stream

    This includes Flow Records carrying the field values for the various observed Flows at each of the Observation Points.

* IPFIX Message

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

* Information Element

  An Information Element is a protocol and encoding-independent description of an attribute that may appear in an IPFIX Record. The IPFIX information model, RFC 5102 [2], defines the base set of Information Elements for IPFIX. The type associated with an Information Element indicates constraints on what it may contain and also determines the valid encoding mechanisms for use in IPFIX.

3. Examples of Flows

Some examples of Flows are listed below. In the IPv4 examples, we use interface addresses in three different 26-bit (/26) subnets. In the IPv6 examples, we use 'mac addr-nn' in the low-order 64 bits to indicate the IEEE MAC (Media Access Control) address of host interface nn.

Example 1: Flow Keys define the different fields by which Flows are distinguished. The different combination of their field values creates unique Flows. If {source IP address, destination IP address, DSCP} are Flow Keys, then all of these are different Flows:

1. {192.0.2.1, 192.0.2.65, 4}
2. {192.0.2.23, 192.0.2.67, 4}
3. {192.0.2.23, 192.0.2.67, 2}
4. {192.0.2.129, 192.0.2.67, 4}

5. {2001:DB8::0:mac-addr-01, 2001:DB8::1:mac-addr-11, 4}
6. {2001:DB8::0:mac-addr-02, 2001:DB8::1:mac-addr-13, 4}
7. {2001:DB8::0:mac-addr-02, 2001:DB8::1:mac-addr-13, 2}
8. {2001:DB8::2:mac-addr-21, 2001:DB8::1:mac-addr-13, 4}

Example 2: A mask function can be applied to all the packets that pass through an Observation Point, in order to aggregate some values. This could be done by defining the set of Flow Keys as {source IP address, destination IP address, DSCP} as in Example 1 above, and applying functions that mask out the source and destination IP addresses (least significant 6 bits for IPv4, 64 bits for IPv6). The eight Flows from Example 1 would now be aggregated into six Flows by merging the Flows 1+2 and 5+6 into single Flows:

1. {192.0.2.0/26, 192.0.2.64/26, 4}
2. {192.0.2.0/26, 192.0.2.64/26, 2}
3. {192.0.2.128/26, 192.0.2.64/26, 4}

4. {2001:DB8::0/64, 2001:DB8::1/64, 4}
5. {2001:DB8::0/64, 2001:DB8::1/64, 2}
6. {2001:DB8::2/64, 2001:DB8::1/64, 4}

Example 3: A filter defined by some Flow Key values can be applied on all packets that pass the Observation Point, in order to select only certain Flows. The filter is defined by choosing fixed values for specific Keys from the packet.

All the packets that go from a customer network 192.0.2.0/26 to another customer network 192.0.2.64/26 with DSCP value of 4 define a Flow. All other combinations don't define a Flow and are not taken

into account. The three Flows from Example 2 would now be reduced to one Flow by filtering out Flows 2 and 3, leaving only Flow 1, {192.0.2.0/26, 192.0.2.64/26, 4}.

Similarly, for the IPv6 packets in the examples above, one could filter out Flows 5 and 6 to leave Flow 4.

The above examples can be thought of as a function F() taking as input {source IP address, destination IP address, DSCP}. The function selects only the packets that satisfy all three of the following conditions:

1. Mask out the least significant 6 bits of source IP address, match against 192.0.2.0.

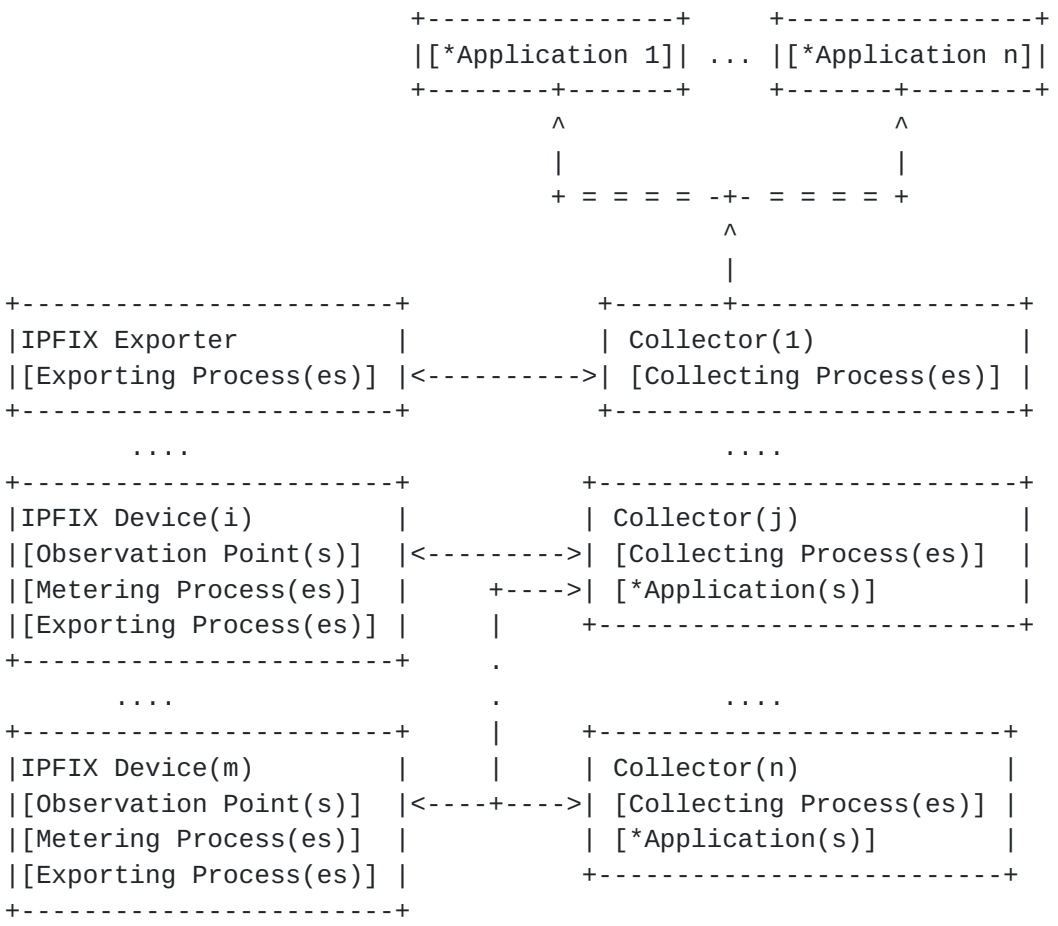
2. Mask out the least significant 6 bits of destination IP address, match against 192.0.2.64.

3. Only accept DSCP value equal to 4.

Depending on the values of {source IP address, destination IP address, DSCP} of the different observed packets, the Metering Process function F() would choose/filter/aggregate different sets of packets, which would create different Flows. For example, for various combinations of values of {source IP address, destination IP address, DSCP}, F(source IP address, destination IP address, DSCP) would result in the definition of one or more Flows.

4. IPFIX Reference Model

The figure below shows the reference model for IPFIX. This figure covers the various possible scenarios that can exist in an IPFIX system.



The various functional components are indicated within brackets []. The functional components within [*] are not part of the IPFIX architecture. The interfaces shown by "<----->" are defined by the IPFIX architecture, but those shown by "<= = = =>" are not.

Figure 1: IPFIX Reference Model

The figure below shows a typical IPFIX Device where the IPFIX components are shown in rectangular boxes.

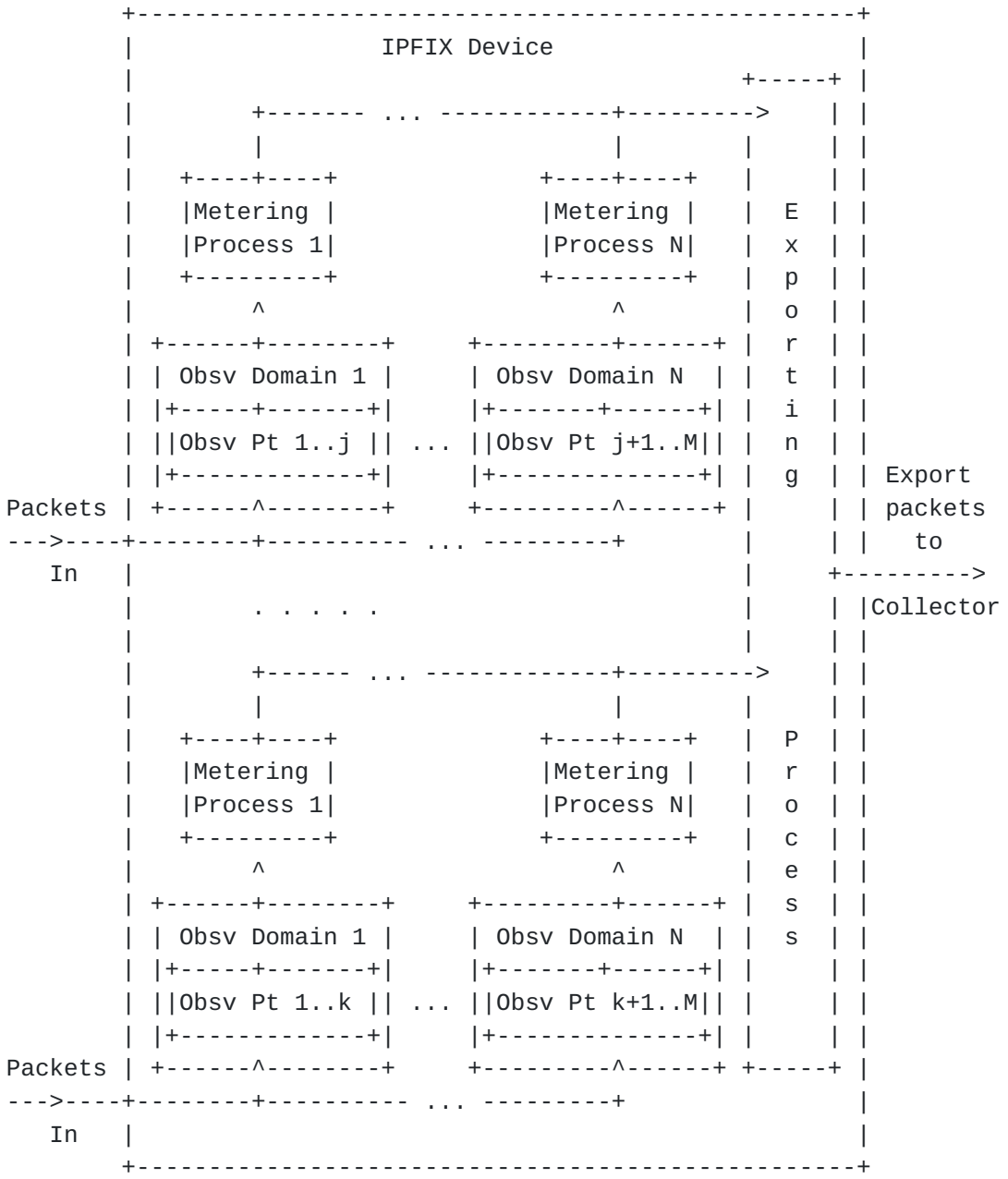


Figure 2: IPFIX Device

5. IPFIX Functional and Logical Blocks

5.1. Metering Process

Every Observation Point in an IPFIX Device, participating in Flow measurements, must be associated with at least one Metering Process. Every packet coming into an Observation Point goes into each of the Metering Processes associated with the Observation Point. Broadly, each Metering Process observes the packets that pass an Observation Point, does timestamping, and classifies the packets into Flow(s) based on the selection criteria.

The Metering Process is a functional block that manages all the Flows generated from an Observation Domain. The typical functions of a Metering Process may include:

- Maintaining database(s) of all the Flow Records from an Observation Domain. This includes creating new Flow Records, updating existing ones, computing Flow Records statistics, deriving further Flow properties, and adding non-Flow-specific information based on the packet treatment (in some cases, fields like AS numbers, router state, etc.)

- Maintaining statistics about the Metering Process itself, such as Flow Records generated, packets observed, etc.

5.1.1. Flow Expiration

A Flow is considered to have expired under the following conditions:

1. If no packets belonging to the Flow have been observed for a certain period of time. This time period should be configurable at the Metering Process, with a minimum value of 0 seconds for immediate expiration. Note that a zero timeout would report a Flow as a sequence of single-packet Flows.

2. If the IPFIX Device experiences resource constraints, a Flow may be prematurely expired (e.g., lack of memory to store Flow Records).

3. For long-running Flows, the Metering Process should expire the Flow on a regular basis or based on some expiration policy. This periodicity or expiration policy should be configurable at the Metering Process. When a long-running Flow is expired, its Flow Record may still be maintained by the Metering Process so that the Metering Process does not need to create a new Flow Record for further observed packets of the same Flow.

5.1.2. Flow Export

The Exporting Process decides when and whether to export an expired Flow. A Flow can be exported because it expired for any of the reasons mentioned in Section 5.1.1, "Flow Expiration". For example: the Exporting Process exports a portion of the expired Flows every 'x' seconds.

For long-lasting Flows, the Exporting Process should export the Flow Records on a regular basis or based on some export policy. This periodicity or export policy should be configurable at the Exporting Process.

5.2. Observation Point

A Flow Record can be better analysed if the Observation Point from which it was measured is known. As such, it is recommended that IPFIX Devices send this information to Collectors. In cases where there is a single Observation Point or where the Observation Point information is not relevant, the Metering Process may choose not to add the Observation Point information to the Flow Records.

5.3. Selection Criteria for Packets

A Metering Process may define rules so that only certain packets within an incoming stream of packets are chosen for measurement at an Observation Point. This may be done by one of the two methods defined below or a combination of them, in either order. A combination of each of these methods can be adopted to select the packets, i.e., one can define a set of methods {F1, S1, F2, S2, S3} executed in a specified sequence at an Observation Point to select particular Flows.

The figure below shows the operations that may be applied as part of a typical Metering Process.

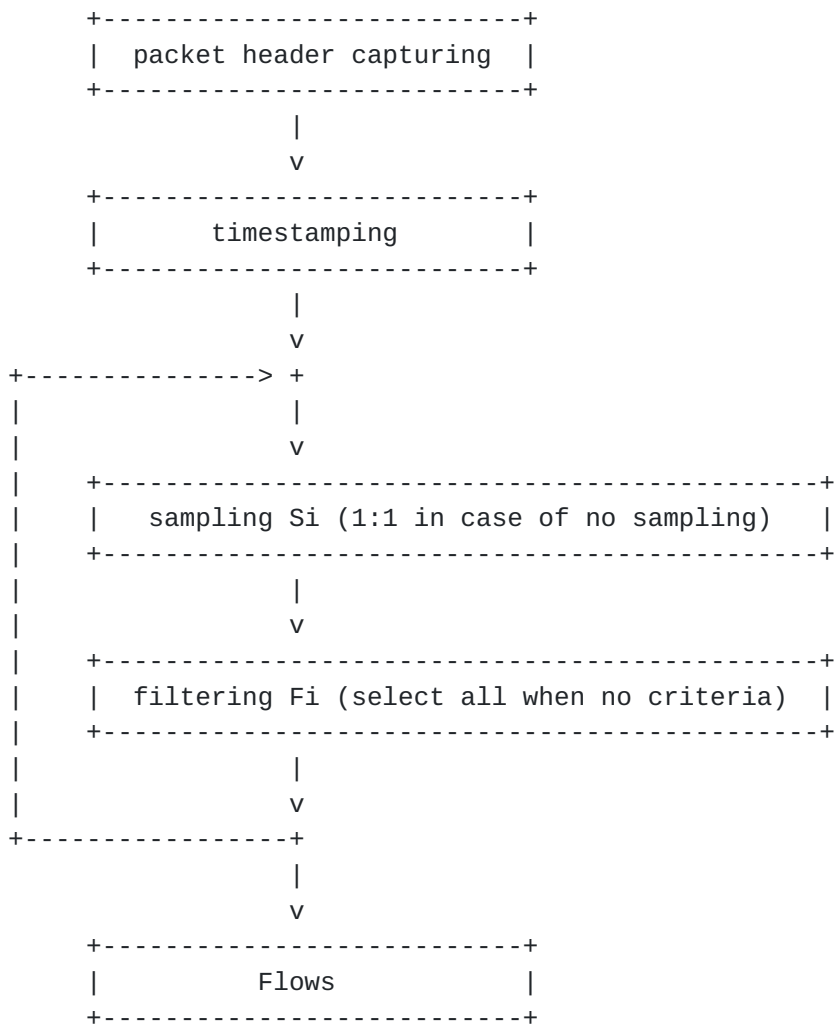


Figure 3: Selection Criteria for Packets

Note that packets could be selected before or after any IP processing, i.e., before there is any IP checksum validation, IP filtering, etc., or after one or more of these steps. This has an impact on what kinds of traffic (or erroneous conditions) IPFIX can observe. It is recommended that packets are selected after their checksums have been verified.

5.3.1. Sampling Functions, Si

A sampling function determines which packets within a stream of incoming packets are selected for measurement, i.e., packets that satisfy the sampling criteria for this Metering Process.

Example: sample every 100th packet that was received at an Observation Point.

Choosing all the packets is a special case where the sampling rate is 1:1.

5.3.2. Filter Functions, Fi

A Filter Function selects only those incoming packets that satisfy a function on fields defined by the packet header fields, fields obtained while doing the packet processing, or properties of the packet itself.

Example: Mask/Match of the fields that define a filter. A filter might be defined as {Protocol == TCP, Destination Port < 1024}.

Several such filters could be used in any sequence to select packets. Note that packets selected by a (sequence of) filter functions may be further classified by other filter functions, i.e., the selected packets may belong to several Flows, any or all of which are exported.

5.4. Observation Domain

The Observation Domain is a logical block that presents a single identity for a group of Observation Points within an IPFIX Device. Each {Observation Point, Metering Process} pair belongs to a single Observation Domain. An IPFIX Device could have multiple Observation Domains, each of which has a subset of the total set of Observation Points in it. Each Observation Domain must carry a unique ID within the context of an IPFIX Device. Note that in the case of multiple Observation Domains, a unique ID per Observation Domain must be transmitted as a parameter to the Exporting Function. That unique ID is referred to as the IPFIX Observation Domain ID.

5.5. Exporting Process

The Exporting Process is the functional block that sends data to one or more IPFIX Collectors using the IPFIX protocol. On one side, the Exporting Process interfaces with Metering Process(es) to get Flow Records; while on the other side, it talks to a Collecting Process on the Collector(s).

There may be additional rules defined within an Observation Domain so that only certain Flow Records are exported. This may be done by either one or a combination of Si and Fi, as described in Section 5.3, "Selection Criteria for Packets".

Example: Only the Flow Records that meet the following selection criteria are exported:

1. All Flow Records whose destination IP address matches {192.0.33.5}.

2. Every other (i.e., sampling rate 1 in 2) Flow Record whose destination IP address matches {192.0.11.30}.

5.6. Collecting Process

Collecting Processes use a Flow Record's Template ID to interpret that Flow Record's Information Elements. To allow this, an IPFIX Exporter must ensure that an IPFIX Collector knows the Template ID for each incoming Flow Record. To interpret incoming Flow Records, an IPFIX Collector may also need to know the function F() that was used by the Metering Process for each Flow.

The functions of the Collecting Process must include:

- Identifying, accepting, and decoding the IPFIX Messages from different <Exporting Process, Observation Domain> pairs.

- Storing the Control Information and Flow Records received from an IPFIX Device.

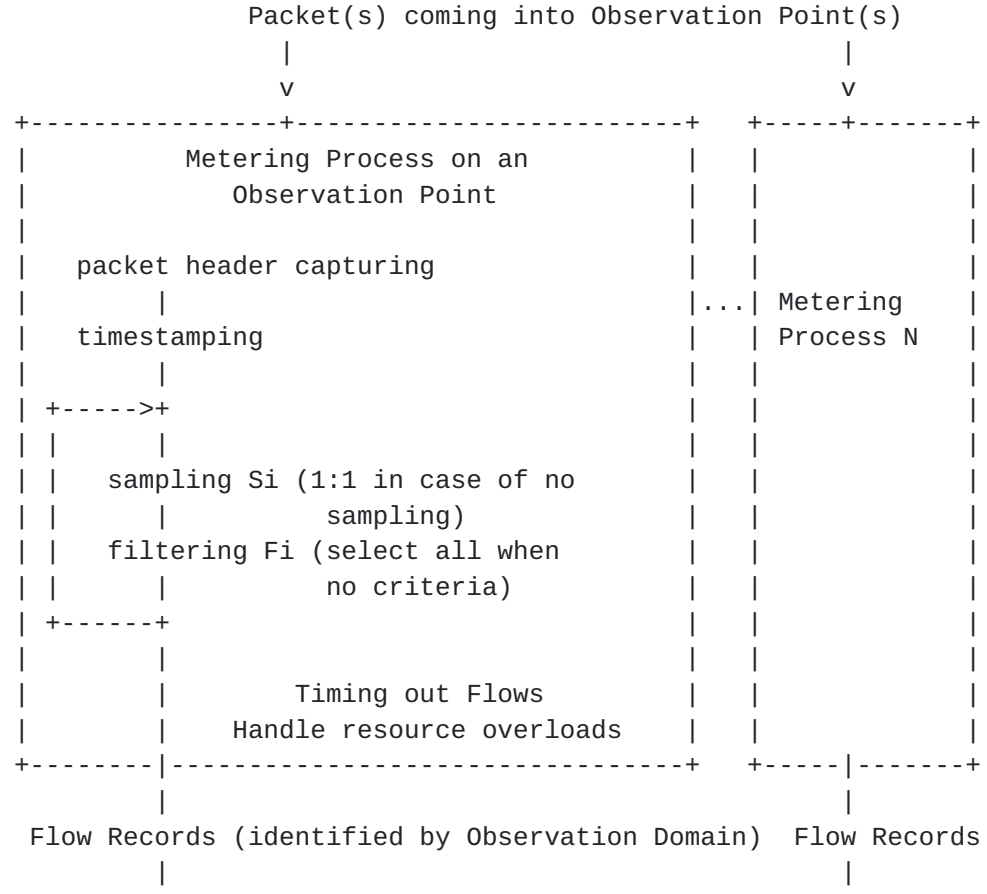
At a high level, the Collecting Process:

1. Receives and stores the Control Information.

2. Decodes and stores the Flow Records using the Control Information.

5.7. Summary

The figure below shows the functions performed in sequence by the various functional blocks in an IPFIX Device.



(*) indicates that the block is optional.

Figure 4: IPFIX Device functional blocks

6. Overview of the IPFIX Protocol

An IPFIX Device consists of a set of cooperating processes that implement the functional blocks described in the previous section. Alternatively, an IPFIX Device can be viewed simply as a network entity that implements the IPFIX protocol. At the IPFIX Device, the protocol functionality resides in the Exporting Process. The IPFIX Exporting Process gets Flow Records from a Metering Process, and sends them to the Collector(s).

At a high level, an IPFIX Device performs the following tasks:

1. Encodes Control Information into Templates.

2. Encodes packets observed at the Observation Points into Flow Records.

3. Packetises the selected Templates and Flow Records into IPFIX Messages.

4. Sends IPFIX Messages to the Collector.

The IPFIX protocol communicates information from an IPFIX Exporter to an IPFIX Collector. That information includes not only Flow Records, but also information about the Metering Process. Such information (referred to as Control Information) includes details of the data fields in Flow Records. It may also include statistics from the Metering Process, such as the number of packets lost (i.e., not metered).

For details of the IPFIX protocol, please refer to RFC 5101 [3].

6.1. Information Model Overview

The IP Flow Information eXport (IPFIX) protocol serves for transmitting information related to measured IP traffic over the Internet. The protocol specification in RFC 5101 [3] defines how Information Elements are transmitted. For Information Elements, it specifies the encoding of a set of basic data types. However, the list of fields that can be transmitted by the protocol, such as Flow attributes (source IP address, number of packets, etc.) and information about the Metering and Exporting Process (packet Observation Point, sampling rate, Flow timeout interval, etc.), is not specified in RFC 5101 [3]. Instead, it is defined in the IPFIX information model in RFC 5102 [2].

The information model provides a complete description of the properties of every IPFIX Information Element. It does this by specifying each element's name, Field Type, data type, etc., and providing a description of each element. Element descriptions give the semantics of the element, i.e., say how it is derived from a Flow or other information available within an IPFIX Device.

6.2. Flow Records

The following rules provide guidelines to be followed while encoding the Flow Records information:

A Flow Record contains enough information so that the Collecting Process can identify the corresponding <Per-Flow Control Information, Configuration Control Information>.

The Exporting Process encodes a given Information Element (as specified in RFC 5102 [2]), based on the encoding standards prescribed by RFC 5101 [3].

6.3. Control Information

The following rules provide guidelines to be followed while encoding the Control Information:

- Per-Flow Control Information should be encoded such that the Collecting Process can capture the structure and semantics of the corresponding Flow data for each of the Flow Records exported by the IPFIX Device.

- Configuration Control Information is conveyed to a Collector so that its Collecting Process can capture the structure and semantics of the corresponding configuration data. The configuration data, which is also Control Information, should carry additional information on the Observation Domain within which the configuration takes effect.

For example, sampling using the same sampling algorithm, say 1 in 100 packets, is configured on two Observation Points O1 and O2. The configuration in this case may be encoded as {ID, observation points (O1,O2), sampling algorithm, interval (1 in 100)}, where ID is the Observation Domain ID for the domain containing O1 and O2. The Observation Domain ID uniquely identifies this configuration, and must be sent within the Flow Records in order to be able to match the right configuration control information.

The Control Information is used by the Collecting Process to:

- Decode and interpret Flow Records.

- Understand the state of the Exporting Process.

Sending Control Information from the Exporting Process in a timely and reliable manner is critical to the proper functioning of the IPFIX Collecting Process. The following approaches may be taken for the export of Control Information:

1. Send all the Control Information pertaining to Flow Records prior to sending the Flow Records themselves. This includes any incremental changes to the definition of the Flow Records.

2. Notify, on a near real-time basis, the state of the IPFIX Device to the Collecting Process. This includes all changes such as a configuration change that affects the Flow behaviour, changes to Exporting Process resources that alter export rates, etc., which the Collector needs to be aware of.

3. Since it is vital that a Collecting Process maintains accurate knowledge of the Exporter's state, the export of the Control Information should be done such that it reaches the Collector reliably. One way to achieve this is to send the Control Information over a reliable transport.

6.4. Reporting Responsibilities

From time to time, an IPFIX Device may not be able to observe all the packets reaching one of its Observation Points. This could occur if a Metering Process finds itself temporarily short of resources. For example, it might run out of packet buffers for IPFIX export.

In such situations, the IPFIX Device should attempt to count the number of packet losses that have occurred, and report them to its Collector(s). If it is not possible to count losses accurately, e.g., when transport layer (i.e., non-IPFIX) errors are detected, the IPFIX Device should report this fact, and perhaps indicate the time period during which some packets might not have been observed.

7. IPFIX Protocol Details

When the IPFIX Working Group was chartered, there were existing common practices in the area of Flow export, for example, NetFlow, CRANE (Common Reliable Accounting for Network Element), LFAP (Light-weight Flow Admission Protocol), RTFM (Real-time Traffic Flow Measurement), etc. IPFIX's charter required the Working Group to consider those existing practices, and select the one that was the closest fit to the IPFIX requirements in RFC 3917 [1]. Additions or modifications would then be made to the selected protocol to fit it exactly into the IPFIX architecture.

7.1. The IPFIX Basis Protocol

The Working Group went through an extensive evaluation of the various existing protocols that were available, weighing the level of compliance with the requirements, and selected one of the candidates as the basis for the IPFIX protocol. For more details of the evaluation process, please see RFC 3955 [6].

In the basis protocol, Flow Records are defined by Templates, where a Template is an ordered set of the Information Elements appearing in a Flow Record, together with their field sizes within those records.

This approach provides the following advantages:

- Using the Template mechanism, new fields can be added to IPFIX Flow Records without changing the structure of the export record format.

- Templates that are sent to the Collecting Process carry structural information about the exported Flow Record fields. Therefore, if the Collector does not understand the semantics of new fields, it can ignore them, but still interpret the Flow Record.

- Because the template mechanism is flexible, it allows the export of only the required fields from the Flows to the Collecting Process. This helps to reduce the exported Flow data volume and possibly provide memory savings at the Exporting Process and Collecting Process. Sending only the required information can also reduce network load.

7.2. IPFIX Protocol on the Collecting Process

The Collecting Process is responsible for:

1. Receiving and decoding Flow Records from the IPFIX Devices.

2. Reporting on the loss of Flow Records sent to the Collecting Process by an IPFIX Exporting Process.

Complete details of the IPFIX protocol are given in RFC 5101 [3].

7.3. Support for Applications

Applications that use the information collected by IPFIX may be Billing or Intrusion Detection sub-systems, etc. These applications may be an integral part of the Collecting Process, or they may be co-located with the Collecting Process. The way by which these applications interface with IPFIX systems to get the desired information is out of scope for this document.

8. Export Models

8.1. Export with Reliable Control Connection

As mentioned in RFC 3917 [1], an IPFIX Device must be able to transport its Control Information and Data Stream over a congestion-aware transport protocol.

If the network in which the IPFIX Device and Collecting Process are located does not guarantee reliability, at least the Control Information should be exported over a reliable transport. The Data Stream may be exported over a reliable or unreliable transport protocol.

Possible transport protocols include:

- SCTP: Supports reliable and unreliable transport.
- TCP: Provides reliable transport only.
- UDP: Provides unreliable transport only. Network operators would need to avoid congestion by keeping traffic within their own administrative domains. For example, one could use a dedicated network (or Ethernet link) to carry IPFIX traffic from Exporter to Collector.

8.2. Collector Failure Detection and Recovery

The transport connection (in the case of a connection-oriented protocol) is pre-configured between the IPFIX Device and the Collector. The IPFIX protocol does not provide any mechanism for configuring the Exporting and Collecting Processes.

Once connected, an IPFIX Collector receives Control Information and uses that information to interpret Flow Records. The IPFIX Device should set a keepalive (e.g., the keepalive timeout in the case of TCP, the HEARTBEAT interval in the case of SCTP) to a sufficiently low value so that it can quickly detect a Collector failure. Note, however, that extremely short keepalive intervals can incorrectly abort the connection during transient periods of congestion. They can also cause some level of additional network load during otherwise idle periods.

Collector failure refers to the crash or restart of the Collecting Process or of the Collector itself. A Collector failure is detected at the IPFIX Device by the break in the connection-oriented transport protocol session; depending on the transport protocol, the connection timeout mechanisms differ. On detecting a keepalive timeout in a

single Collector scenario, the IPFIX Device should stop sending Flow Records to the Collector and try to reestablish the transport connection. If Collecting Process failover is supported by the Exporting Process, backup session(s) may be opened in advance, and Control Information sent to the failover Collecting Process.

There could be one or more secondary Collectors with priority assigned to them. The primary Collector crash is detected at the IPFIX Device. On detecting loss of connectivity, the IPFIX Device opens a Data Stream with the secondary Collector of the next highest priority. If that secondary was not opened in advance, both the Control Information and Data Stream must be sent to it. That Collector might then become the primary, or the Exporting Process might try to reestablish the original session.

8.3. Collector Redundancy

Configuring redundant Collectors is an alternative to configuring backup Collectors. In this model, all Collectors simultaneously receive the Control Information and Data Streams. Multiple {Control Information, Data Stream} pairs could be sent, each to a different Collector, from the same IPFIX Device. Since the IPFIX protocol requires a congestion-aware transport, achieving redundancy using multicast is not an option.

9. IPFIX Flow Collection in Special Situations

An IPFIX Device can generate, receive, and/or alter two special types of traffic, which are listed below.

Tunnel traffic:

> The IPFIX Device could be the head, midpoint, or end-point of a tunnel. In such cases, the IPFIX Device could be handling Generic Routing Encapsulation (GRE) [8], IPinIP [7], or Layer Two Tunneling Protocol version 3 [9] traffic.

VPN traffic:

> The IPFIX Device could be a provider-edge device that receives traffic from customer sites belonging to different Virtual Private Networks.

Similarly, IPFIX could be implemented on devices which perform one or more of the following special services:

- Explicitly drop packets. For example, a device that provides firewall service drops packets based on some administrative policy.

- Alter the values of fields used as IPFIX Flow Keys of interest. For example, a device that provides NAT service can change the source and/or destination IP address.

In cases such as those listed above, there should be clear guidelines as to:

- How and when to classify the packets as Flows in the IPFIX Device.

- If multiple encapsulations are used to define Flows, how to convey the same fields (e.g., IP address) in different layers.

- How to differentiate Flows based on different private domains. For example, overlapping IP addresses in Layer-3 VPNs.

- What extra information needs to be exported so that the Collector can make a clear interpretation of the received Flow Records.

10. Security Considerations

Flow information can be used for various purposes, such as usage-based accounting, traffic profiling, traffic engineering, and intrusion detection. The security requirements may differ significantly for such applications. To be able to satisfy the security needs of various IPFIX users, an IPFIX system must provide different levels of security protection.

10.1. Data Security

IPFIX data comprises Control Information and Data Streams generated by the IPFIX Device.

The IPFIX data may exist in both the IPFIX Device and the Collector. In addition, the data is also transferred on the wire from the IPFIX Device to the Collector when it is exported. To provide security, the data should be protected from common network attacks.

The protection of IPFIX data within the end system (IPFIX Device and Collector) is out of scope for this document. It is assumed that the end system operator will provide adequate security for the IPFIX data.

The IPFIX architecture must allow different levels of protection to the IPFIX data on the wire. Wherever security functions are required, it is recommended that users should leverage lower layers using either TLS or DTLS (Datagram Transport Layer Security), if these can successfully satisfy the security requirement of IPFIX data protection.

To protect the data on the wire, three levels of granularity should be supported; these are described in the following subsections.

10.1.1. Host-Based Security

Security may not be required when the transport between the IPFIX Device and the Collector is perceived as safe. This option allows the protocol to run most efficiently without extra overhead, and an IPFIX system must support it.

10.1.2. Authentication-Only

Authentication-only protection provides IPFIX users with the assurance of data integrity and authenticity. The data exchanged between the IPFIX Device and the Collector is protected by an authentication signature. Any modification of the IPFIX data will be detected by the recipient, resulting in the discarding of the received data. However, the authentication-only option doesn't offer data confidentiality.

The IPFIX user should not use authentication-only when sensitive or confidential information is being exchanged. An IPFIX solution should support this option. The authentication-only option should provide replay attack protection. Some means to achieve this level of security are:

- Encapsulating Security Payload (with a null encryption algorithm)
- Transport Layer Security (with a null encryption algorithm)
- IP Authentication Header

10.1.3. Encryption

Data encryption provides the best protection for IPFIX data. The IPFIX data is encrypted at the sender, and only the intended recipient can decrypt and have access to the data. This option must be used when the transport between the IPFIX Device and the Collector is unsafe, and the IPFIX data needs to be protected. It is

recommended that the underlying transport layer's security functions be used for this purpose. Some means to achieve this level of security are:

- Encapsulating Security Payload

- Transport Layer Security Protocol

The data encryption option adds overhead to the IPFIX data transfer. It may limit the rate that an Exporter can report its Flow Records to the Collector, due to the resource requirement for running encryption.

10.2. IPFIX End-Point Authentication

It is important to make sure that the IPFIX Device is talking to the "right" Collector rather than to a masquerading Collector. The same logic also holds true from the Collector's point of view, i.e., it may want to make sure it is collecting the Flow Records from the "right" IPFIX Device. An IPFIX system should allow the end-point authentication capability so that either one-way or mutual authentication can be performed between the IPFIX Device and Collector.

The IPFIX architecture should use any existing transport protection protocols, such as TLS, to fulfil the authentication requirement.

10.3. IPFIX Overload

An IPFIX Device could become overloaded under various conditions. This may be because of exhaustion of internal resources used for Flow generation and/or export. Such overloading may cause loss of data from the Exporting Process, either from lack of export bandwidth (possibly caused by an unusually high number of observed Flows) or from network congestion in the path from Exporter to Collector.

IPFIX Collectors should be able to detect the loss of exported Flow Records, and should at least record the number of lost Flow Records.

10.3.1. Denial-of-Service (DoS) Attack Prevention

Since one of the potential usages for IPFIX is for intrusion detection, it is important for the IPFIX architecture to support some kind of DoS resistance.

10.3.1.1. Network under Attack

The network itself may be under attack, resulting in an overwhelming number of IPFIX Messages. An IPFIX system should try to capture as much information as possible. However, when a large number of IPFIX Messages are generated in a short period of time, the IPFIX system may become overloaded.

10.3.1.2. Generic DoS Attack on the IPFIX Device and Collector

The IPFIX Device and Collector may be subject to generic DoS attacks, just as any system on any open network. These types of attacks are not IPFIX specific. Preventing and responding to such types of attacks are out of the scope of this document.

10.3.1.3. IPFIX-Specific DoS Attack

There are some specific attacks on the IPFIX portion of the IPFIX Device or Collector:

- The attacker could overwhelm the Collector with spoofed IPFIX Export packets. One way to solve this problem is to periodically synchronise the sequence numbers of the Flow Records between the Exporting and Collecting Processes.

- The attacker could provide false reports to the Collector by sending spoofed packets.

The problems mentioned above can be solved to a large extent if the control packets are encrypted both ways, thereby providing more information that the Collector could use to identify and ignore spoofed data packets.

11. IANA Considerations

The IPFIX Architecture, as set out in this document, has two sets of assigned numbers, as outlined in the following subsections.

11.1. Numbers Used in the Protocol

IPFIX Messages, as described in RFC 5101 [3], 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.

Values for the IPFIX Version Number and the IPFIX Set ID, together with the considerations for assigning them, are defined in RFC 5101 [3].

11.2. Numbers Used in the Information Model

Fields of the IPFIX protocol carry information about traffic measurement. They are modelled as elements of the IPFIX information model RFC 5102 [2]. Each Information Element describes a field that may appear in an IPFIX Message. Within an IPFIX Message, the field type is indicated by its Field Type.

Values for the IPFIX Information Element IDs, together with the considerations for assigning them, are defined in RFC 5102 [2].

12. Acknowledgements

The document editors wish to thank all the people contributing to the discussion of this document on the mailing list, and the design teams for many valuable comments. In particular, the following made significant contributions:

Tanja Zseby
Paul Calato
Dave Plonka
Jeffrey Meyer
K.C.Norseth
Vamsi Valluri
Cliff Wang
Ram Gopal
Jc Martin
Carter Bullard
Reinaldo Penno
Simon Leinen
Kevin Zhang
Paul Aitken
Brian Trammell

Special thanks to Dave Plonka for the multiple thorough reviews.

13. References

13.1. Normative References

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

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

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

[4] Zseby, T., Boschi, E., Brownlee, N., and B. Claise, "IPFIX Applicability", RFC 5472, March 2009.

13.2. Informative References

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

[6] Leinen, S., "Evaluation of Candidate Protocols for IP Flow Information Export (IPFIX)", RFC 3955, October 2004.

[7] Simpson, W., "IP in IP Tunneling", RFC 1853, October 1995.

[8] Farinacci, D., Li, T., Hanks, S., Meyer, D., and P. Traina, "Generic Routing Encapsulation (GRE)", RFC 2784, March 2000.

[9] Lau, J., Townsley, M., and I. Goyret, "Layer Two Tunneling Protocol - Version 3 (L2TPv3)", RFC 3931, March 2005.

Authors' Addresses

Ganesh Sadasivan
Rohati Systems
1192 Borregas Ave.
Sunnyvale, CA 94089
USA

EMail: gsadasiv@rohati.com

Nevil Brownlee
CAIDA | The University of Auckland
Private Bag 92019
Auckland 1142
New Zealand

Phone: +64 9 373 7599 x88941
EMail: n.brownlee@auckland.ac.nz

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

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

Juergen Quittek
NEC Laboratories Europe, NEC Europe Ltd.
Kurfuersten-Anlage 36
Heidelberg 69115
Germany

Phone: +49 6221 4342-115
EMail: quittek@nw.neclab.eu
URI: http://www.neclab.eu/