IPFIX Working Group
Internet-Draft
Intended Status: Standards Track
Expires: July, 2014

P. Aitken
Cisco Systems, Inc.

31 Dec 2013

Reporting Unobserved Fields in IPFIX
draft-aitken-ipfix-unobserved-fields-02

Status of this Memo

Copyright Notice

Abstract

The IPFIX protocol is designed to export information about observations, and lacks a method for reporting that observations are unavailable. This document discusses several methods for reporting when fields are unavailable, reviews the advantages and disadvantage of each, and recommends methods which should be used.

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

TODO: would it be useful to define two new fields for reporting the start and end time of a period during which no observations were made?

TODO: how do these methods work with structured data and MIBs?

1. Introduction

The IPFIX information model [RFC7012] contains a wide variety of fields [IANA-IPFIX], which are not always present in all traffic. For example, ICMP type and code. Indeed, some fields are mutually exclusive. For example, IPv4 / IPv6 address fields; UDP / TCP port numbers.

When an IPFIX Metering Process monitors a field, it only reports whenever appropriate traffic is seen. ie, whenever the Observed Traffic Stream [RFC5476] contains the field to be metered. No output is generated whenever the monitored field is not present.

The IPFIX protocol lacks a method for the Metering Process to actively express that although certain fields were being monitored, no relevant observations were made, that a particular field is not applicable, or that a value could not be calculated for a derived field. Therefore the Collecting Process cannot know whether the Metering Process was actively monitoring the field, and can only infer from the lack of export that no relevant observations were made.

Further, when a Metering Process monitors a combination of fields, some may be present while others are not. Therefore the Metering Process may observe values for some fields, though not for others.

The IPFIX Protocol [RFC7011] requires the Exporting Process to employ a number of templates in order to export these combinations, using one template for each observed combination of fields. Since the number of templates required grows exponentially with the number of field combinations, the templates can quickly become unmanageable. Indeed, just a few sets of mutually exclusive fields are sufficient to exhaust the template number space.

Clearly the IPFIX protocol [RFC7011] requires a method for the Metering Process to actively express that although fields were being monitored, no relevant observations were made.

Note that there are several cases where an observation may not be available for a given field:

1. Unavailable / Not applicable:

   The monitored field was not present in, or not applicable to, the observed traffic.
   eg, when RTP SSRC is requested for a TCP flow.

2. Not Calculated:

   Although the required fields exist, something is preventing the calculation from occurring. For example, not enough data has been collected to calculate a TCP round-trip time.

This document discusses various potential methods for reporting such unobserved fields, reviews the advantages and disadvantages of each, and recommends suitable methods as an extension to the IPFIX Protocol [RFC7011].

2. Terminology

IPFIX-specific terminology used in this document is defined in

Section 2 of the IPFIX protocol specification [RFC7011]. As in [RFC7011], these IPFIX-specific terms have the first letter of a word capitalized when used in this document.

2.1 New Terminology

Unobserved Field

> A field which a Metering Process is metering, but for which no traffic has been seen or no data is available.

3. Potential Methods

This section discusses and evaluates various possible methods for reporting Unobserved Fields.

3.1 Zero-valued counters

This method exports counters with a value of zero to indicate that no traffic was observed.

For example, the following data records use zero-valued counters to indicate that no traffic was observed from the specified source or sent to the specified destination:

- sourceIPv4Address = n.n.n.n, packetTotalCount = 0
- destinationIPv4Prefix = n.n.n.n, packetTotalCount = 0
- bgpSourceAsNumber = nnnn, octetTotalCount = 0
- destinationTransportPort = pppp, octetTotalCount = 0
- protocolIdentifier = P, octetTotalCount = 0

Clearly this only works for specific addresses, address prefixes, Autonomous systems, port numbers, protocols. The method is not suitable for exhaustively reporting all addresses, prefixes, Autonomous Systems, ports, or protocols from which no traffic was sent or received.

Therefore, while this is a useful method, it addresses a different problem. It does not meet the requirement of being able to report unobserved fields and is therefore not a solution.

3.2 Multiple Templates

The NetFlow v9 [RFC3954] and IPFIX [RFC7011] protocols are both template based. Templates express which fields are present in the exported Data Records.

When no value has been observed for a particular field, a new template is generated without that corresponding Information Element. Thus the Unobserved Field is simply omitted from the export. ie, no values are exported for unobserved fields.

While this prevents an incorrect or misleading value from being exported for the Unobserved Field, it may require a great many Templates to be created. The number may soon become unmanageable, or may exceed the Template number space - especially when the Metering Process is metering a large number of mutually-exclusive fields.

The IPFIX template number space is a little less than 16 bits (256 through 65535 = 65280 templates), so the combinations of 16 different unobserved fields will exhaust the number space.

Eg, IPv4 and IPv6 traffic require separate templates, in order to report either sourceIPv4Address and destinationIPv4Address, or sourceIPv6Address and destinationIPv6Address.

Again, udpSourcePort and udpDestinationPort are mutually exclusive with tcpSourcePort and tcpDestinationPort. Although this may be mitigated by using the generic sourceTransportPort and destinationTransportPort Information Elements together with the protocolIdentifier, not all traffic is port based. Eg, these port fields are mutually exclusive with icmpTypeCodeIPv4 and icmpTypeCodeIPv6.

A final example is the MPLS label stack. Depending how many MPLS labels are present in the Observed Traffic Stream, up to ten different templates may be required, containing the ten MPLS

label stack Information Elements, mplsTopLabelStackSection through mplsLabelStackSection10.

The main issue with this method is that since no data is exported about unobserved fields, the Collecting Process cannot tell whether the field was not being observed by the Metering Process, or was being observed but no relevant traffic was seen.

Eg, if mplsLabelStackSection is not exported, the Collecting Process is unable to determine whether MPLS traffic is being actively monitored and no relevant traffic was seen, or MPLS traffic is not being monitored.

Therefore this method does not meet the requirement of being able to report unobserved fields and is therefore not a solution.

3.3 CommonProperties

Common Properties divides Data Records into a core part in which the fields are always observed, and additional parts according to which other fields are observed. These are exported using the method described in [RFC5473].

One template is required to express which fields are present in the core, and one Template is required for each combination of additional fields. Since multiple templates are required, the number of Templates may soon become unmanageable or may exceed the Template number space as discussed in section 3.2 above.

Although Common Properties [RFC5473] isn't specified for NetFlow v9 [RFC3954], there is no technical reason preventing this.

The main issue with this method is that again, like the Multiple Templates case above, no data is exported about Unobserved Fields - so the Collecting Process cannot tell whether the field

was not being observed, or was being observed but no relevant traffic was seen.

Therefore this method does not meet the requirement of being able to report unobserved fields.

3.4 Default Values

With this method, a single Template specifies all the fields which the Metering Process was asked to observe, regardless of

whether values were observed and are available for each of the fields. Fields for which no value was observed, or for which the value is unavailable, are exported with a default value. The default value can be of several kinds as discussed below.

Note that multiple default values are required if it's necessary to distinguish between the "not applicable" and "not required" cases. In general, both cases may be represented by a single value.

Since only one template is used, this scheme is trivial to implement and works for both NetFlow v9 [RFC3954] and IPFIX.

Specific default values are discussed in the following sections.

3.4.1 Default of Zero

All unobserved fields are exported with the value zero. This has the advantage that neither the Exporting Process nor the Collecting Process needs any extra knowledge about the field.

However, if zero is a valid value for the field, it will be impossible to distinguish an unobserved field from a real observation. Eg, when reporting the number of lost packets, packetsLost = 0 seems to indicate that no traffic was lost, when it may be intended to indicate that there is no relevant information to report. Even IP addresses of 0.0.0.0 and 0::0 are valid representations. And zero is a valid value for icmpTypeCodeIPv4 (indicating echo reply).

Therefore, although this method reports unobserved fields, it's not always possible to determine whether or not the field was observed. Therefore this method is not a suitable solution.

3.4.2 Default of all-ones

The "Default of all-ones" method is similar to the "Default of Zero" method discussed above, except that the all-ones value (0xFF, 0xFFFF, 0xFFFFFFFF, etc) is used for each Unobserved Field.

Such values are often, though not always, reserved and therefore may more clearly indicate whether or not the field was observed.

However, this method has the additional disadvantage that the default value for Unobserved Fields changes with the size of the field. Eg, 255 for 8-bit fields, 65535 for 16-bit fields, etc.

Additionally, field sorting is impacted without additional logic

to recognise the "all-ones" value, since "unobserved" appears as the topmost value.

For this reason, and because these values are not always reserved, this method is not a suitable solution.

3.4.3 General default value

This method is similar to the "Default of Zero" and "Default of all-ones" methods discussed above, except that another non-zero, non-0xFF...FF default value is used for each Unobserved Field. Eg, a repeating pattern of ASCII space (0x20), or a hex number such as "0xD15AB1ED".

However, it seems impossible to choose a value which would never appear in a real observation, both for existing Information Elements and for new Information Elements defined in future.

Eg, although "0XD15AB1EDD15AB1ED" may be quite unlikely, that's not enough to give a robust indication. Further, "0XD15A" is possible (eg, port 53594) and "0xD1" has a 1-in-256 chance of incorrectly indicating that a one-octet field was unobserved.

Therefore this method is not a suitable solution.

3.4.4 Field-specific default values

Each field is provided with a special "unobserved" value, which is outside the normal range of observed values. This value may vary from field to field.

When no value is observed for the field, it's exported with the "unobserved" value.

Eg, to report that the flow direction is not relevant to the current flow, the flowDirection Information Element could be exported with any value other than 0 (ingress) or 1 (egress).

Similarly, to report that the IP version is not relevant to the current flow, the ipVersion Information Element could be

exported with a value between 10 and 15 (see [IANA-VERSION]).

Since the "unobserved" value is outwith the normal range of values for the field, it is possible to distinguish an unobserved field from a real observation.

However, both the Exporting Process and the Collecting Process need extra knowledge about each individual field.

Further, not all Information Elements have a suitable value. Eg, counter, time, and process ID Information Elements may use their entire range of bits.

Therefore this method suffers from the same problem as the other "Default Value" methods above, and is not a suitable solution.

3.5 "observedFieldsIndicator" bitfield

With this method, a single template contains Information Elements for all the fields which the Metering Process was asked to observe.

Additionally, the Template contains an "observedFieldsIndicator" bitfield similar to the "flowKeyIndicator" (see [IANA-IPFIX] 173), in that each bit corresponds to one Information Element in the flow record. Each bit in this field indicates whether or not a value was observed for the corresponding Information Element within the Data Record.

For each Data Record, the Collecting Process examines the "observedFieldsIndicator" bitfield to discover which fields were observed and which were unobserved within that Data Record. Unobserved fields may therefore be exported with any value, since they will be disregarded.

Since it uses only one template, this scheme is trivial to implement and works for both NetFlow v9 [RFC3954] and IPFIX.

However, mediators MUST understand the "observedFieldsIndicator" bitfield and correctly interpret it. eg, if the mediator is aggregating Data Records, it MUST pay attention to the bitfield in order to disregard data for unobserved fields. Additionally, if fields are added to or removed from the Flow Record, bits in the bitfield must be shifted accordingly. Therefore this requires changes to IPFIX record processing.

Note that if the "observedFieldsIndicator" bitfield is sent in an IPFIX Options Record, it expresses which fields are valid in that Options Data Record. It's not possible to use option scoping to report the "observedFieldsIndicator" bitfield for any other Record.

Although this method clearly reports unobserved fields, it's limited to a simple binary indication of whether or not a value was observed. It's not possible to give a reason, or to distinguish "Unavailable", "Not applicable" and "not calculated" as discussed in section 1.

Provided that the simple boolean indication is sufficient, this method provides a good solution.

This method requires a new "observedFieldsIndicator" Information Element, as specified in section 5, "New Information Elements".

3.6 Length of Zero

This method exports a single Template containing Information

Elements for all the fields which the Metering Process was asked to observe. Fields for which no value was observed are exported using IPFIX variable-length encoding [RFC7011], with a length of zero. Therefore unobserved fields are actively indicated.

Fields which may be unobserved must be anticipated ahead of time and specified in the Template using IPFIX variable-length encoding [RFC7011].

While this method only requires a single Template, it doesn't work for NetFlow v9 export [RFC3954] since NetFlow v9 doesn't support variable-length encoding.

It also does not address the not-applicable versus not-calculated case discussed in section 3.5, which may be needed for some fields.

However, provided that the simple boolean indication is sufficient, and especially when variable-length encoding is already being used for the field, this method provides a good solution.

3.7 Size field

With this method, a single Template contains Information Elements for all the fields which the Metering Process was asked to observe.

In addition, a "size" or "count" field is added to the Template, indicating how many of the fields are valid.

Eg, the Template contains mplsTopLabelStackSection, mplsLabelStackSection2, ..., mplsLabelStackSection10.

Additionally, mplsLabelStackDepth is provided, indicating how many of the MPLS label elements are valid.

Clearly this method is field-specific. Instead, the solution should use a structured data list as discussed in section 3.8 below. Therefore this method is not recommended.

3.8 Structure data lists

With this method, a single template contains Information Elements for all the fields which the Metering Process was asked to observe. A list is appended to each Data Record to provide more information about the fields in the Template.

Several types of list are possible as described below.

3.8.1 Status list

A new Information Element defines the status of each field. Eg, observed, unavailable, not applicable, not calculated. One status is provided per Information Element.

A status list is encoded using a basicList as described in [RFC6313], and the list is appended to the Data Record.

This method is similar to the "observedFieldsIndicator" bitfield discussed in section 3.5, with the advantage that more information is available about each field.

However, this method suffers from the same mediator issue. ie, the list must be understood by mediators, and must be modified when a mediator adds fields to, or removes fields from, the Data Record.

With one octet per status, the Data Record is not overly burdened.

While this method offers some advantage over the "observedFieldsIndicator" bitfield discussed in section 3.5, it is not recommended.

3.8.2 Observed field list

A list of informationElementIndex, indicating which of the elements in the Template contains valid values, is appended to each Data Record.

Elements which appear in the Template but not in the list were not observed, not applicable, or could not be calculated.

Since informationElementIndex is 16 bits long, this method produces a list which is twice as long as that in section 3.8.1 above.

Since the order of the fields in the list is unimportant, when a mediator adds fields to the Data Record, the list can simply be extended. Therefore the mediator issue is mitigated to some extent.

However, it's not possible to tell why a particular Information Element is not available.

Since the "Combined" method discussed in section 3.8.3 below offers a better solution, this method is not recommended.

3.8.3 Combined field and status list

This method combines the status and field list from sections 3.8.1 and 3.8.2, by exporting a subTemplateList containing {informationElementIndex, status} pairs.

Therefore this method produces a list which is thrice as long as that in section 3.8.1 above. Additionally, a further template is required to encode the {informationElementIndex, status} pair.

The status makes it possible for the Collecting Process to understand why a particular Information Element is not available.

Since the order of the fields in the list is unimportant, when a mediator adds fields to the Data Record, the list can simply be extended. Therefore the mediator issue is mitigated to some extent.

This method is the most flexible and informative of all the structured data solutions. However, it incurs a penalty of an additional template to export the subTemplateList. Therefore this method is not recommended.

Not-TODO: could be implemented as two parallel basicLists, or a basicList of a new "Element+Status" IE.

4. Conclusion

Several methods of encoding "unobserved" fields have been presented. Each has pros and cons.

The only methods which satisfactorily meet the requirements are the "observedFieldsIndicator" bitfield in section 3.5, and the "Length of Zero" method in section 3.6.

These methods may be combined in order to report when no value is available for a given export field.

In the general case, the "observedFieldsIndicator" bitfield method specified in section 3.5 should be used.

However, when IPFIX variable-length encoding can be used, or is already being used, the "Length of Zero" method specified in section 3.6 may be used.

Therefore this document specifies an extension to the IPFIX protocol [RFC7011], such that a variable-length encoding with a length of zero indicates that no value was available for the corresponding Information Element.

The ability to indicate Unobserved Fields conveys an additional benefit: the ability for the Metering Process to indicate that it has begun to monitor a new flow, but does not yet have anything to export. Therefore, all the fields are unobserved.

5. New Information Elements

This document defines the following new IPFIX Information Elements:

observedFieldsIndicator

Description:
This bit field indicates which of the Information Elements within a Data Record have been observed. Each bit of the observedFieldsIndicator represents an Information Element in the Data Record with the n-th bit representing the n-th Information Element.
A bit set to value 1 indicates that the corresponding Information Element was observed and contains a valid value. A bit set to value 0 indicates that the corresponding Information Element was not observed and contains an invalid value. The Information Element value SHOULD be set to zero, and MUST be disregarded by the Collecting Process.
Information Elements which have no observedFieldsIndicator bit MUST contain a valid value.
If the Data Record contains more than 64 Information Elements, the corresponding Template SHOULD be designed such that all unobserved fields are among the first 64 Information Elements, because the observedFieldsIndicator only contains 64 bits. If the Data Record contains less than 64 Information Elements, then the bits in the observedFieldsIndicator for which no corresponding Information Element exists MUST have the value 0.

Abstract Data Type: unsigned64
Data Type Semantics: flags
ElementId: TBD1

6. Security Considerations

No additional security considerations are introduced in this document. The same security considerations as for the IPFIX protocol [RFC7011] apply.

7. IANA Considerations

Additional Information Elements to be allocated in the [IANA-IPFIX] registry per section 5, "New Information Elements."

8. References

8.1 Normative References

[RFC7011] Claise, B., Ed., Trammell, B., Ed., and P. Aitken, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information", STD 77, RFC 7011, September 2013.

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

8.2 Informative References

[RFC7012] Claise, B., Ed., and B. Trammell, Ed., "Information Model for IP Flow Information Export (IPFIX)", RFC 7012, September 2013.

[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.

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

[RFC6313] Claise, B., Dhandapani, G., Aitken, P., and S. Yates, "Export of Structured Data in IP Flow Information Export (IPFIX)", RFC 6313, July 2011.

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

[IANA-VERSION]
http://www.iana.org/assignments/version-numbers/version-numbers.xml

9. Acknowledgements

Thanks to Aamer Akhter and Luca Deri for initial review and feedback, to Andrew Johnson for the lists method, and to Jan Novak, Andrew Johnson, Colin McDowall, and Dana Blair for reviewing the draft and providing feedback.

10. Author's Address

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

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