Exporting MIB Variables using the IPFIX Protocol

draft-ietf-ipfix-mib-variable-export-02

Juergen Schoenwaelder, Srikar B S, Benoit Claise, Paul Aitken

86th IETF Meeting, Orlando, 2013

Overview

- This document specifies a way to complement IPFIX Flow Records with Management Base (MIB) objects, avoiding the need to define new IPFIX Information Elements for existing Management Information Base objects that are already fully specified.

Review: -01 changes

- -01 introduced Extended Field Specifiers
  - to address an issue on the mailing list (http://www.ietf.org/mail-archive/web/ipfix/current/msg06347.html)
  - to resolve multiple open issues in -00

-02 changes 1/2

- Reworked all the examples in section 6 to show how the new EFSF is used.
- Added new “MIB Context Identifier” IE with new “mibContext” data type.
- Numerous small corrections, clarifications, and improvements to the text.

-02 changes 2/2

- Confusion between “IEindex” and “informationElementIndex” resolved by renaming to “previousIE index”:

Type	Extension Name	Extension Description
0	Reserved	Reserved
1	MIB OID	The extension contains a MIB Object Identifier.
2	MIB index	The extension contains an additional MIB as an index.
3	IE index	The extension contains an additional IPFIX Information Element as an index.
4	**previousIE index**	The extension contains the informationElementIndex of a previous Information Element in the Flow Record as an index.
5	MIB Instance Identifier index	The extension contains a MIB Instance Identifier as an index.

Table 1: Extension Types

Resolved Issues

- Should the Field Length be zero, and extension 1 data length carry the length? Conflicts with "unobserved fields".

- No; the field length says how long the MIB will be; the ext1 length is the ext length, not the field length.

- Revise or delete section 5.4 "Indices Considerations".

  In a previous version, indices could repeat information. This is no longer the case, so section deleted.

Open Issues 1/2

- Replace RFC5101 references with 5101bis.
- Replace RFC5102 references into 5102bis or IANA-IPFIX.
- "timestamps, exporters, and other animals" -> see the mailing list.
- The value of the MIB OID acting as an index may not be of fixed length and may have no default length, for example the OID can be of type string or type MIB OID.
- Some TODO in the XML version:
  - write section 6.7: "Indexed MIB Objects with a mix of MIB OID and IPFIX Information Element"
  - write section 6.10: "Using MIB Objects with IPFIX Structured Data"
  - write section 6.11: "Using IPFIX Structured Data to group the index MIB and indices"

Open Issues 2/2

- RFC 5610: explain what needs to be updated.
- ID to name mappings? Use this for an example in section 5.
- What does this mean? : "(Consider the counter synchronization issue, non-key info should be static)".
- (JS) Do we need to add something about the contextEngineID and contextName?
- (JS) Inacio's figure: send email to the mailing list.
- Unobserved fields could be reported in EFSF format, ie by an "(un)observed" extension.
- Tidy up the XML.

The message is: there's still a lot of work to do.

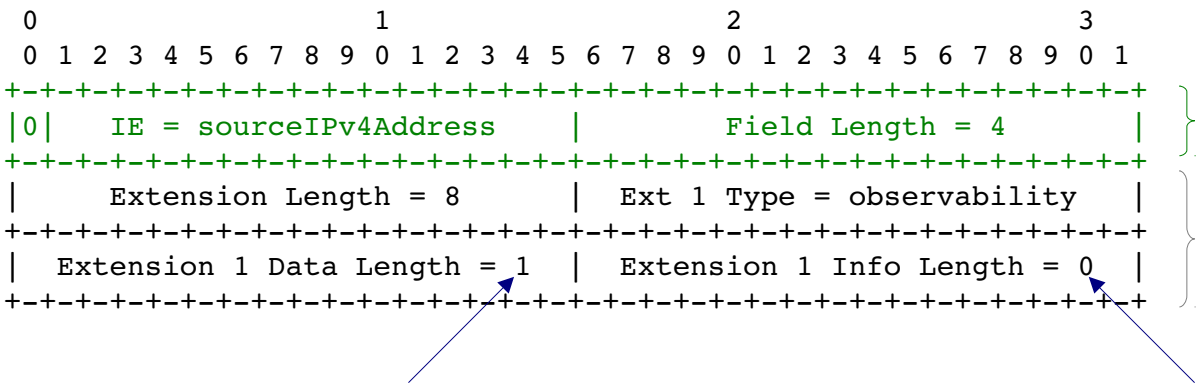
New Issue: unobserved fields

- Recall that draft-aitken-ipfix-unobserved-fields-01 proposes several methods for reporting when fields are unavailable, not applicable, or not calculated.

  Eg:
  - no UDP ports for TCP traffic
  - no IPv4 addresses for IPv6 traffic
  - no packets to calculate RTT

- Another method of reporting unobserved fields is to use the EFSF from the MIB export draft to report an "(un)observed" property, with values "observed", "not available", "not applicable".

EFSF: example for unobserved fields

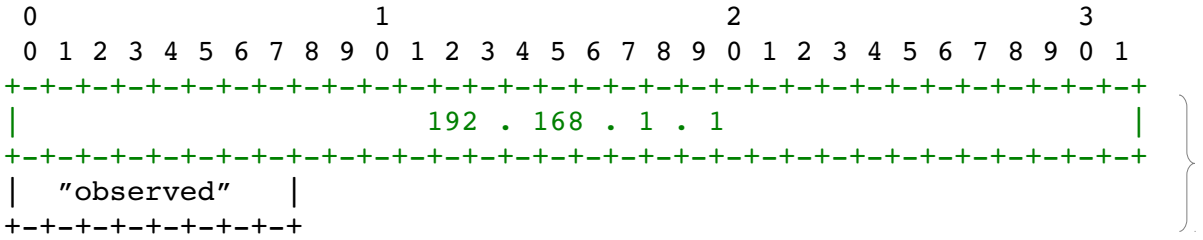


Standard Field Specifier

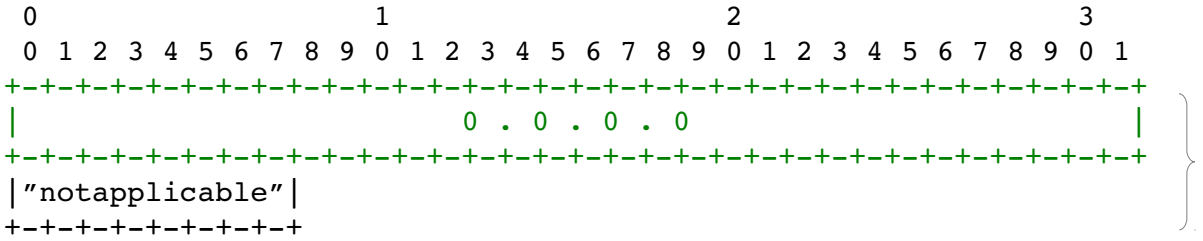
"Observability" Extension

Contributes 1 extra octet to Data Records.

No further information within the template



data record for IPv4 traffic



data record for non-IPv4 traffic

EFSF: what else? 1/2

Type	Length	Value	Details
Key	0	key / non-key	Key fields distinguish one flow from another.
Non-key behaviour	1	min / max / average / first / last	How the value of a non-key field was determined.
Direction	0	ingress / egress	Whether traffic was ingress or egress.
Observation point	1	OP ID	Location where the traffic was observed. Eg, interface, NAT process, QOS process...
Pre / Post	0	pre / post	Whether the observation was made before (pre) or after (post) packet treatment.
Biflow direction	0	forward / reverse	Forward versus reverse fields, without the clumsy RFC 5103 PEN mechanism.
Biflow strategy	0	initiator / responder	Which side of the biflow is which?
Counter semantics	0	delta / total	deltaCounter versus totalCounter sematics, without requiring duplicate fields.
Aggregation count	4	original / aggregated	How many flows were aggregated together. A value of “1” indicates an unaggregated flow.
Time	1	start / end	Start and end timestamp, without requiring duplicate fields.
MIB	N	MIB OID	The OID of the MIB being exported.
Observability	1	observed / not available / not applicable	Indicates whether a value was observed, and why not.

EFSF: what else? 2/2

Type	Length	Value	Details
Offset	1	packet offset	The offset of the captured data within a packet section.
Autonomous system	0	peer / origin	Whether the AS ID is from a peer or origin.
Interface type	1	physical / logical / channelised / virtual	The interface type.
Error type	0	absolute / relative	Whether the error is absolute or relative.
Error amount	4	amount of error	
Hash options	tbd	tbd	tbd
Name	N	(string)	informationElementName
Range	N	X, Y	informationElementRangeBegin, informationElementRangeEnd
Semantics	1	(semantics)	informationElementSemantics
Units	1	(units)	informationElementUnits
Index	N	Field index	Field index, eg encapsulation layer.
Enterprise-specific	4	PEN	Indicates the PEN for ES elements.

EFSF: use cases 1/3 : IE equivalence

- Today we export “ingressInterface” and “egressInterface” and assume that “interfaceName” applies equally to both.
- Since interfaceName is directionless, use EFSF with direction, index, and name properties:
- Data record:
  interface.{dir=ingress} = 123
  interface.{dir=egress} = 456
- Option record:
  interface.{index=123}.{name=“eth1”}
  interface.{index=456}.{name=“eth2”}

EFSF: use cases 2/3 : index and encaps

- Indexing multiple instances of an IE within a data record, eg MPLS label stack:
  MPLSlabel.{stackLevel=1} = xxxx
  MPLSlabel.{stackLevel=2} = yyyy
  MPLSlabel.{stackLevel=3} = zzzz

- Reporting traffic hierarchy and inner headers.
  eg, report IPv6 encapsulated in IPv4:

  sourceIPv4address.{encapsLevel=1}
  destinationIPv4address.{encapsLevel=1}

  sourceIPv6address.{encapsLevel=2}
  destinationIPv6address.{encapsLevel=2}

EFSF: use cases 3/3

- Application export:
  app.{id} = 123
  app.{engine} = NBAR
  app.{name} = “http”
  app.{subapp}.{browser} = chrome
  app.{subapp}.{browser}.{version} = 25.0.1364.172
  app.{subapp}.{url} = cisco.com

- MIB export:
  mib.{oid} = 1.3.6.1.2.1.2.2.1.1
  mib.{index} = 5

- (Un)observed fields:
  f.{observed} = observed / not available / not applicable

EFSF: conclusion

- EFSF is an orthogonal mechanism to the IPFIX information model.

- Solves several issues:
  - MIB export
  - indexing, hierarchical, and positional elements
  - inter-relationship between elements (min/max/average, first/last, ingress/egress, pre/post)
  - biflow (RFC5103)
  - exporting type (RFC5610)

- Should this be a new WG item?

- Avoid delaying the MIB export draft!

Exporting MIB Variables using the IPFIX Protocol

draft-ietf-ipfix-mib-variable-export-02

Juergen Schoenwaelder, Srikar B S, Benoit Claise, Paul Aitken

86th IETF Meeting, Orlando, 2013