

Applying IPFIX to Network Measurement and Management

presented by

Brian Trammell

Monday 12 May 2014

RIPE 68 — Warsaw, Poland

Acknowledgments

- mPlane
- Benoit Claise
 - who co-authored/presented a version of this tutorial at IETF 87 in Berlin
- Elisa Boschi
 - who co-authored/presented a version of this tutorial at the 2008 Internet Measurement Conference (IMC)
- and of course the document authors, reviewers, chairs, and other contributors of the IPFIX Working Group, who did all the actual work.

Once over lightly

WHAT is IPFIX?

What is IPFIX?

- “IP Flow Information eXport”
- IETF Standard (STD 77)
- a unidirectional **protocol** for data export;
- a **data format** providing efficient record-level self-description for this protocol;
 - applicable to any collection with large numbers of records sharing similar structures
- and an **information model** providing the vocabulary for this data format.
 - applicable to most measurement/logging tasks at transport and network layers, extensible beyond.

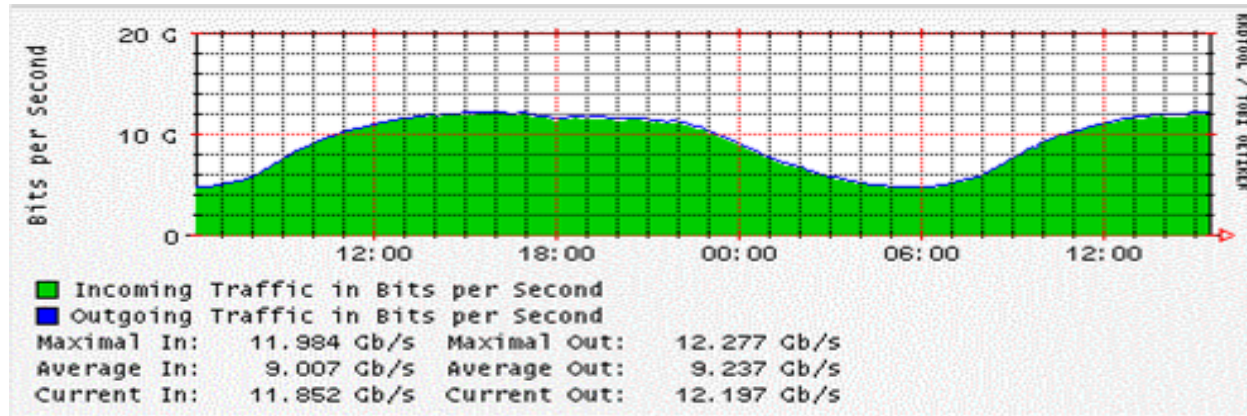
Today's Agenda

- A History of Flow Measurement and IPFIX
- Architecture
- Protocol Structures: IEs and Data Format
- Protocol Dynamics and Transport
- Cross-Area Applications

How we got here

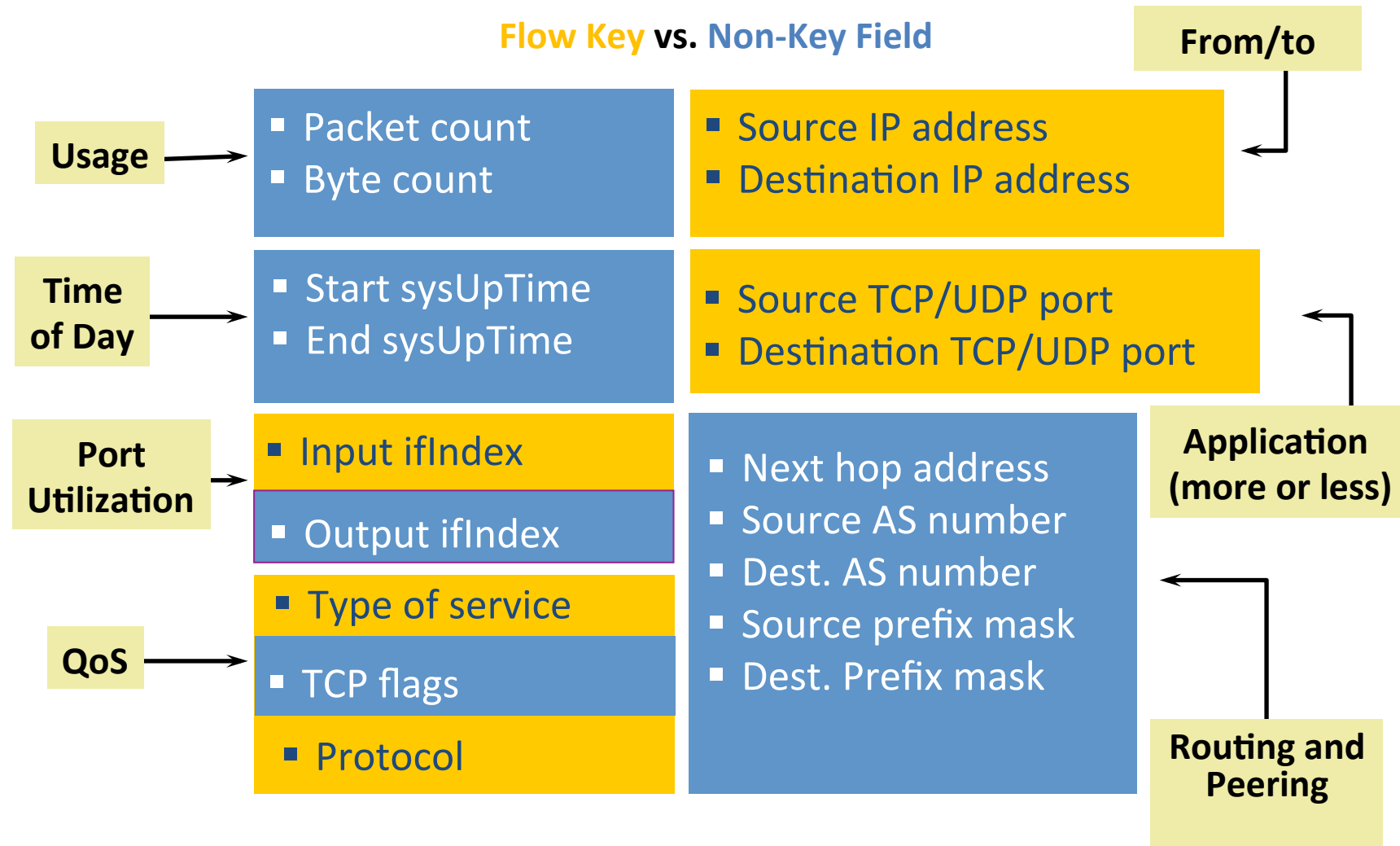
A HISTORY OF FLOW MEASUREMENT

Monitoring Background



- 5 minutes interface counters polling (typically MRTG)
- Potentially RMON event/alarm for threshold notifications
- In some case, the EXPRESSION MIB (RFC 2982)
 - A new MIB variable for link utilization
- Troubleshooting: packet capture
- Flow monitoring between interface counters and packet capture

NetFlow Version 5 Flow Format



NetFlow Cache Example

1. Create and update flows in NetFlow cache

SrcIf	SrcIPadd	DstIf	DstIPadd	Protocol	TOS	Flgs	Pkts	Src Port	Src Msk	Src AS	Dst Port	Dst Msk	Dst AS	NextHop	Bytes /Pkt	Active	Idle
Fa1/0	173.100.21.2	Fa0/0	10.0.227.12	11	80	10	11000	00A2	/24	5	00A2	/24	15	10.0.23.2	1528	1745	4
Fa1/0	173.100.3.2	Fa0/0	10.0.227.12	6	40	0	2491	15	/26	196	15	/24	15	10.0.23.2	740	41.5	1
Fa1/0	173.100.20.2	Fa0/0	10.0.227.12	11	80	10	10000	00A1	/24	180	00A1	/24	15	10.0.23.2	1428	1145.5	3
Fa1/0	173.100.6.2	Fa0/0	10.0.227.12	6	40	0	2210	19	/30	180	19	/24	15	10.0.23.2	1040	24.5	14

2. Expiration

- Inactive timer expired (15 sec is default)
- Active timer expired (30 min is default)
- NetFlow cache is full (oldest flows are expired)
- RST or FIN TCP flag

SrcIf	SrcIPadd	DstIf	DstIPadd	Protocol	TOS	Flgs	Pkts	Src Port	Src Msk	Src AS	Dst Port	Dst Msk	Dst AS	NextHop	Bytes/ Pkt	Active	Idle
Fa1/0	173.100.21.2	Fa0/0	10.0.227.12	11	80	10	11000	00A2	/24	5	00A2	/24	15	10.0.23.2	1528	1800	4

3. Aggregation

4. Export version

Non-aggregated flows—export **version 5 or 9**

5. Transport protocol (UDP, SCTP)

Export Packet

Header

Payload (Flows)

E.g., Protocol-Port Aggregation Scheme Becomes

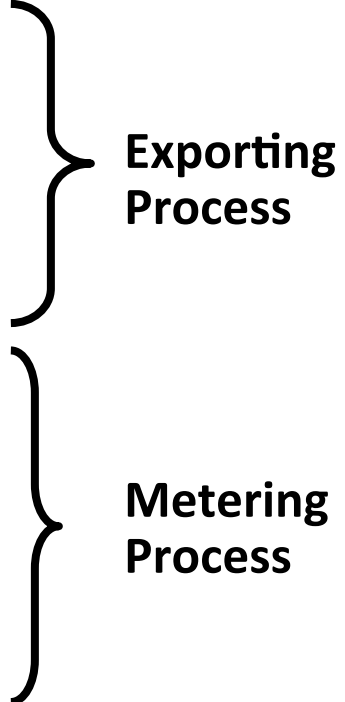
Protocol	Pkts	SrcPort	DstPort	Bytes/Pkt
11	11000	00A2	00A2	1528

Aggregated Flows—Export **Version 8 or 9**

Flow Monitoring History

- NetFlow, Cisco proprietary technology
 - 20th anniversary in 2015
- First attempt to standardize a flow technology at the IETF:
 - Realtime Traffic Flow Measurement (RTFM)
 - From 1997 to 1997
 - RFC 2063, RFC 2064, RFC 2123, RFC 2720, RFC 2721, RFC 2722, RFC 2723, RFC 2724
 - <https://datatracker.ietf.org/wg/rtfm/>

Lesson learned: Extensibility and Flexibility Requirements

- Traditional NetFlow with v5 or v8 NetFlow export
 - New requirements: build something flexible and extensible
 - Phase One: NetFlow Version 9
 - Advantages: extensibility
 - Integrate new technologies/data types quicker (MPLS, IPv6, BGP next hop, etc.)
 - Integrate new aggregations quicker
 - Phase Two: Flexible NetFlow
 - Advantages: cache and export content flexibility
 - User selection of flow keys
 - User definition of the records
- 
- Exporting Process**
- Metering Process**

Flexible Flow Record: Key Fields

Flow
Sampler ID
Direction
Class ID
Interface
Input
Output
Layer 2
Source VLAN
Dest VLAN
Dot1q VLAN
Dot1q priority
Source MAC address
Destination MAC address

IPv4	
IP (Source or Destination)	Payload Size
Prefix (Source or Destination)	Packet Section (Header)
Mask (Source or Destination)	Packet Section (Payload)
Minimum-Mask (Source or Destination)	TTL
Protocol	Options bitmap
Fragmentation Flags	Version
Fragmentation Offset	Precedence
Identification	DSCP
Header Length	TOS
Total Length	

IPv6	
IP (Source or Destination)	Payload Size
Prefix (Source or Destination)	Packet Section (Header)
Mask (Source or Destination)	Packet Section (Payload)
Minimum-Mask (Source or Destination)	DSCP
Protocol	Extension Headers
Traffic Class	Hop-Limit
Flow Label	Length
Option Header	Next-header
Header Length	Version
Payload Length	

Flexible Flow Record: Key Fields

Routing	Transport		Application
src or dest AS	Destination Port	TCP Flag: ACK	Application ID
Peer AS	Source Port	TCP Flag: CWR	
Traffic Index	ICMP Code	TCP Flag: ECE	Multicast
Forwarding Status	ICMP Type	TCP Flag: FIN	
IGP Next Hop	IGMP Type*	TCP Flag: PSH	
BGP Next Hop	TCP ACK Number	TCP Flag: RST	
Input VRF Name	TCP Header Length	TCP Flag: SYN	
	TCP Sequence Number	TCP Flag: URG	
	TCP Window-Size	UDP Message Length	
	TCP Source Port	UDP Source Port	
	TCP Destination Port	UDP Destination Port	
	TCP Urgent Pointer	RTP SSRC	Replication Factor*
			RPF Check Drop*
			Is-Multicast

Flexible Flow Record: Non-Key Fields

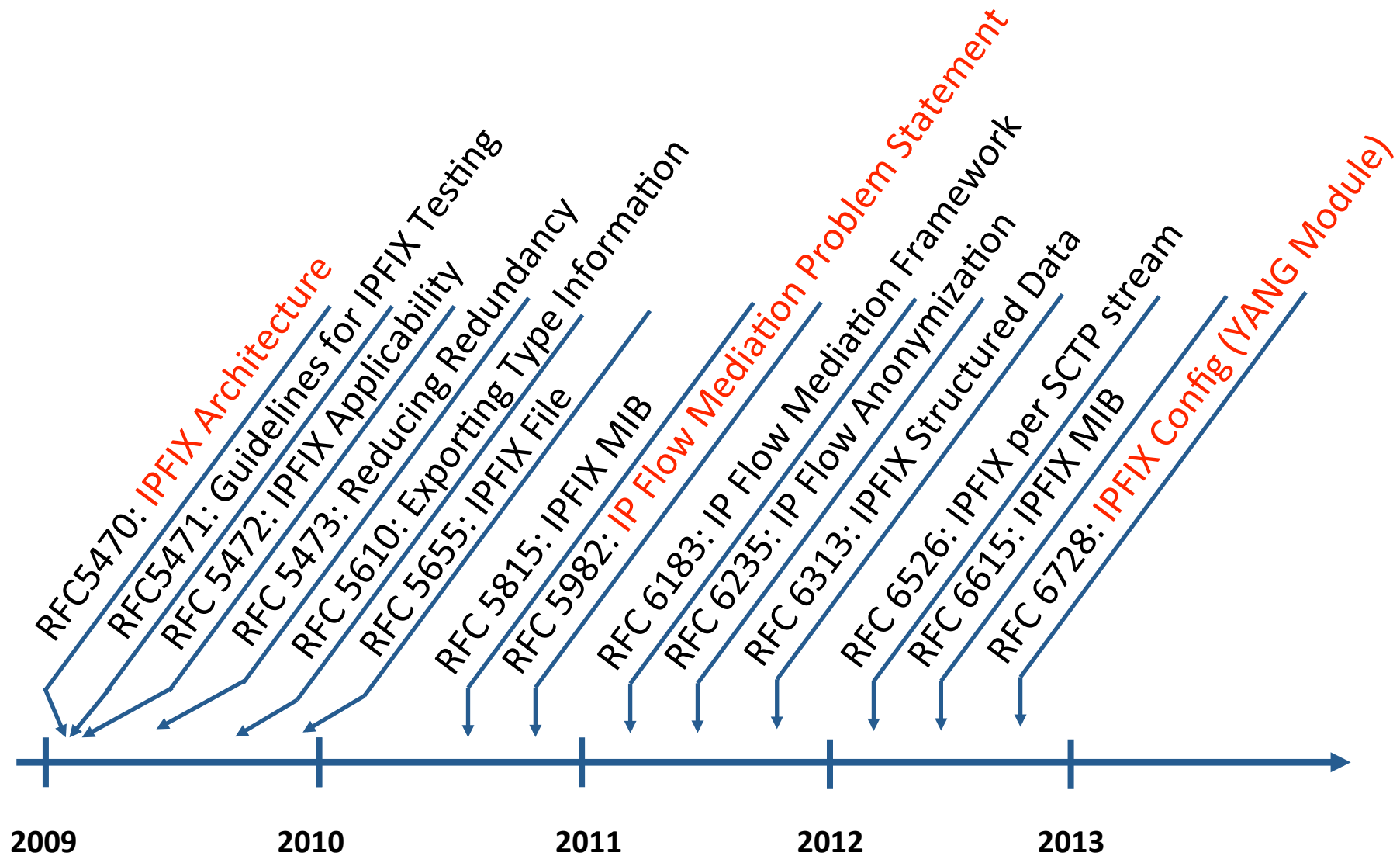
Counters	Timestamp	IPv4	IPv4 and IPv6
Bytes	sysUpTime First Packet	Total Length Minimum	Total Length Minimum
Bytes Long	sysUpTime First Packet	Total Length Maximum	Total Length Maximum
Bytes Square Sum	Absolute first packet	TTL Minimum	
Bytes Square Sum Long	Absolute last packet	TTL Maximum	
Packets			
Packets Long			
Bytes replicated			
Bytes replicated Long			
Packets replicated			
Packets Replicated Long			

- Plus any of the potential “key” fields: will be the value from the first packet in the flow

IPFIX History: The Early Years

- 2001: BOF at IETF 51
- 2002-2005: Discussion of requirements and candidate protocols
 - RFC 3917: Requirements
 - RFC 3954: Specification of NetFlow V9
 - RFC 3955: Evaluation of candidate protocols
 - CRANE, LFAP, Diameter, sFlow, NetFlow (V9)
- 2005-2008: Transport discussion
 - UDP vs TCP vs SCTP (vs DCCP) / IPsec vs TLS
- 2008: RFC 5101, RFC 5102 Published

IPFIX History: Extension and Expansion



IPFIX History: Completion (2013)

- RFC 7011, 7012 published
 - Internet Standard versions of protocol and information model specification
- RFC 7013: Information Element guidelines
 - IANA IE registry now normative reference
 - IE registry maintained without RFCs
- Currently wrapping up work in the IETF IPFIX Working Group

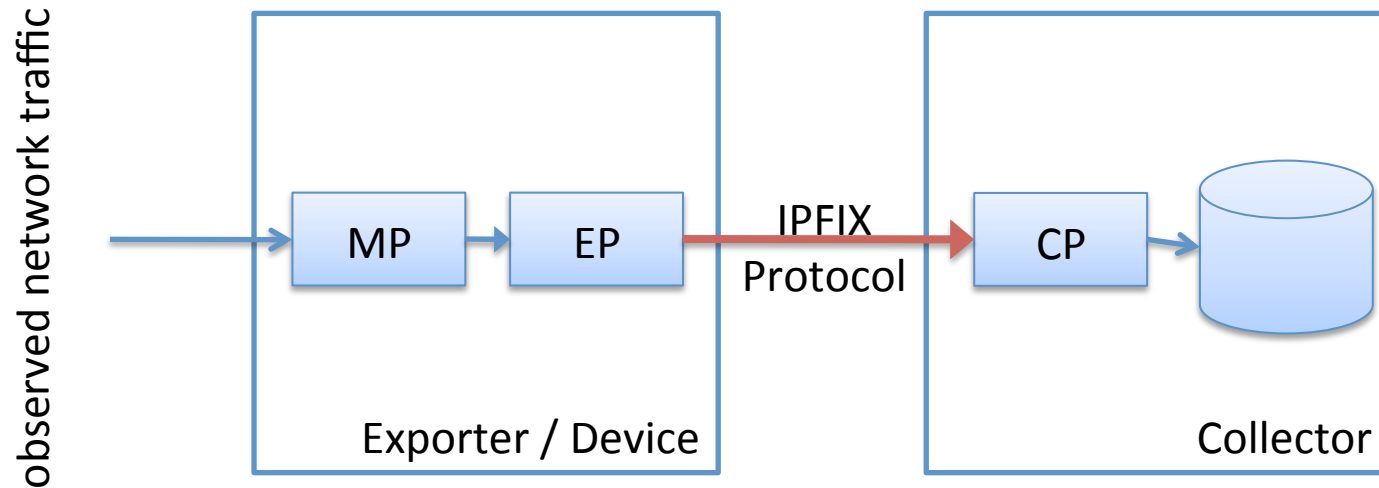
IPFIX Devices and Processes

ARCHITECTURE

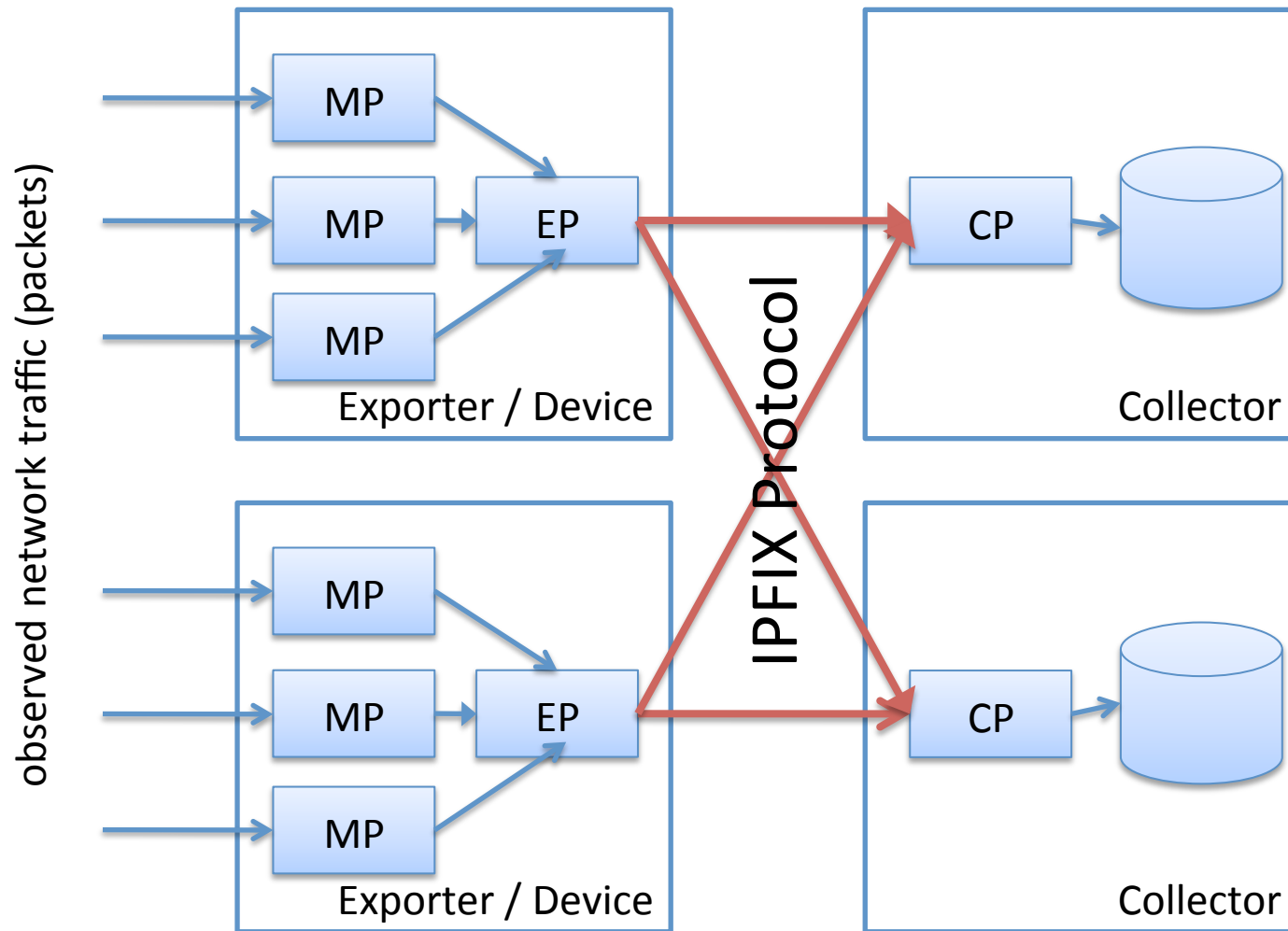
Architecture Terminology (RFC 5470)

- **Metering Process (*MP*)**: generates *Flow Records* from packets at an *Observation Point*. Performs packet capture; timestamping, sampling, and classification of flows; maintains flows in some internal data structure; and passes complete Flow Records to an...
- **Exporting Process (*EP*)**: Sends Flow Records via IPFIX from one or more Metering Processes to one or more Collecting Processes.
- **Collecting Process (*CP*)**: Receives Flow Records via IPFIX from one or more Exporting Processes.

Simple Architecture



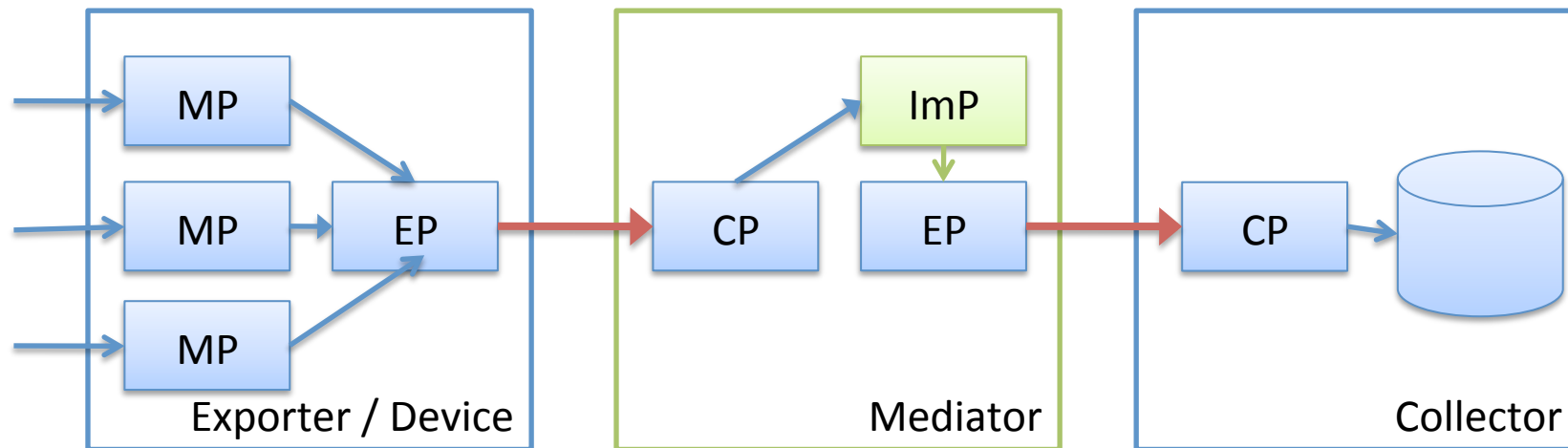
General Architecture



IPFIX Mediators

- *Mediators* collect, transform, and re-export IPFIX Message streams.
- Allow federation of IPFIX
- Framework in RFC 6183, protocol considerations in RFC 7119.
- *Intermediate Processes (ImP)* transform data:
 - Anonymization (RFC 6235)
 - Aggregation (RFC 7015)
 - Filtering, proxying, mux/demux, protocol translation, etc.

observed network traffic (packets)



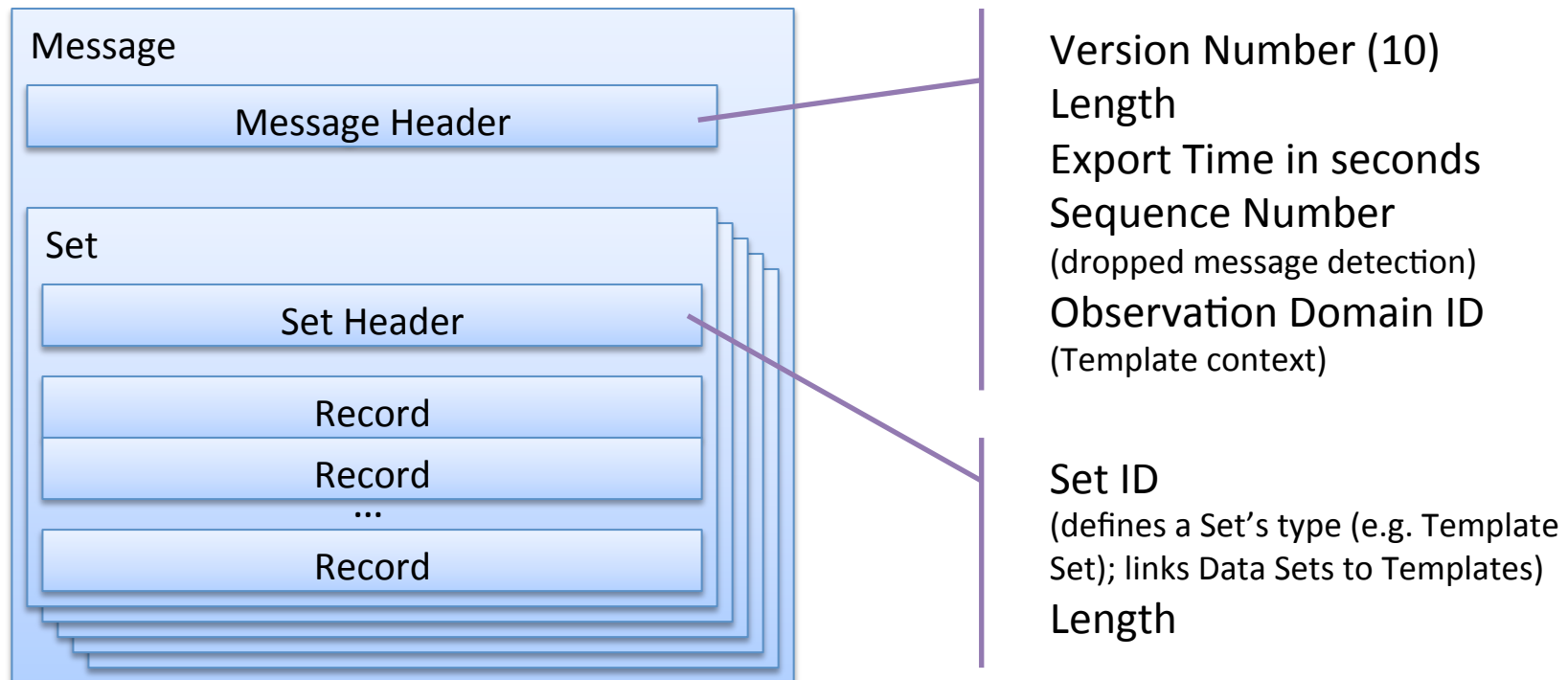
IPFIX at Rest: Messages, Sets, Templates, and Information Elements

PROTOCOL STRUCTURES

Protocol Terminology Overview

- IPFIX transports flow data in *(IPFIX) Messages*.
- A Message contains a *Message Header* and one or more *Sets*.
- A Set contains a *Set Header* and may be one of:
 - a *Template Set*, containing *Template Records*;
 - an *Options Template Set*, containing *Options Template Records*; or
 - a *Data Set*, containing *Data Records*.
 - The structure of these Data Records is described by a corresponding Template or Options Template.

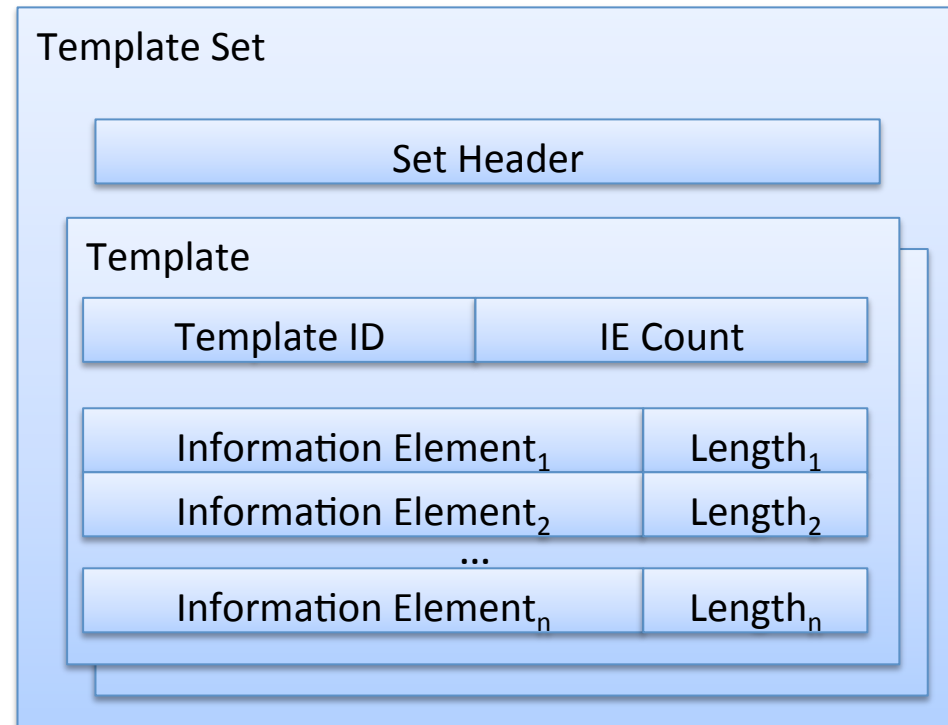
Message Structure



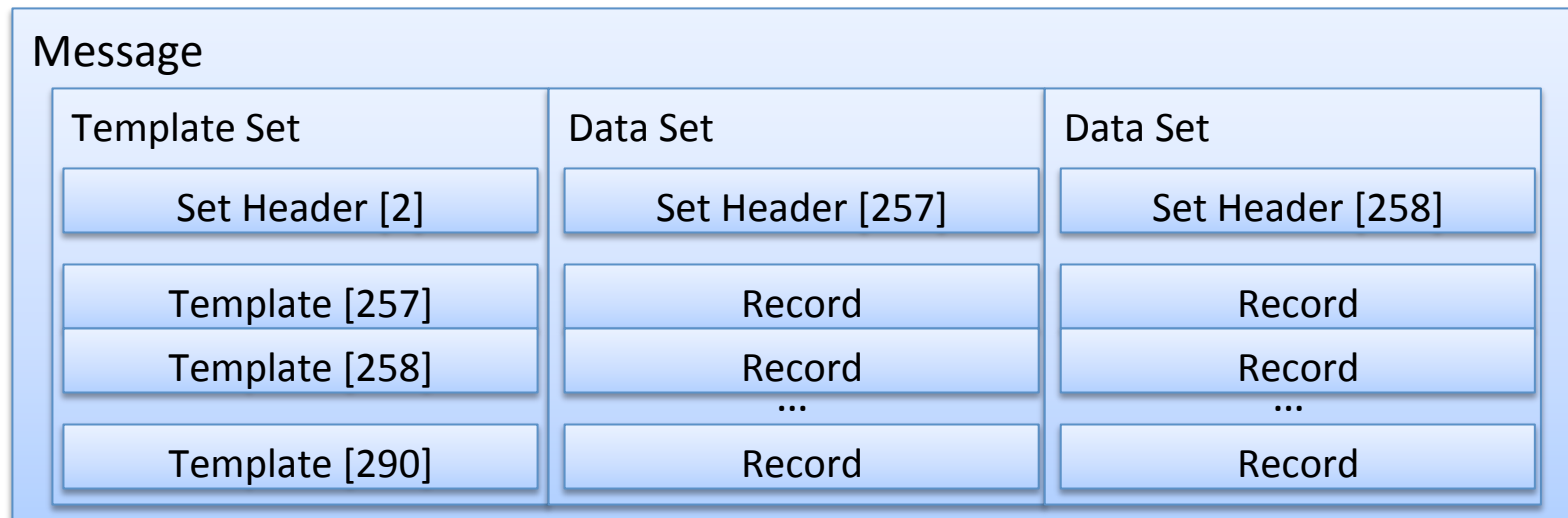
Templates and Information Elements

- A *Template* describes the structure of Data Records within a Data Set.
- Templates identified by *Template ID*,
 - which corresponds to *Set ID* in the Set Header of the Data Set.
- Templates are composed of {*Information Element (IE)*, length} pairs.
- IEs provide field type information for Templates.

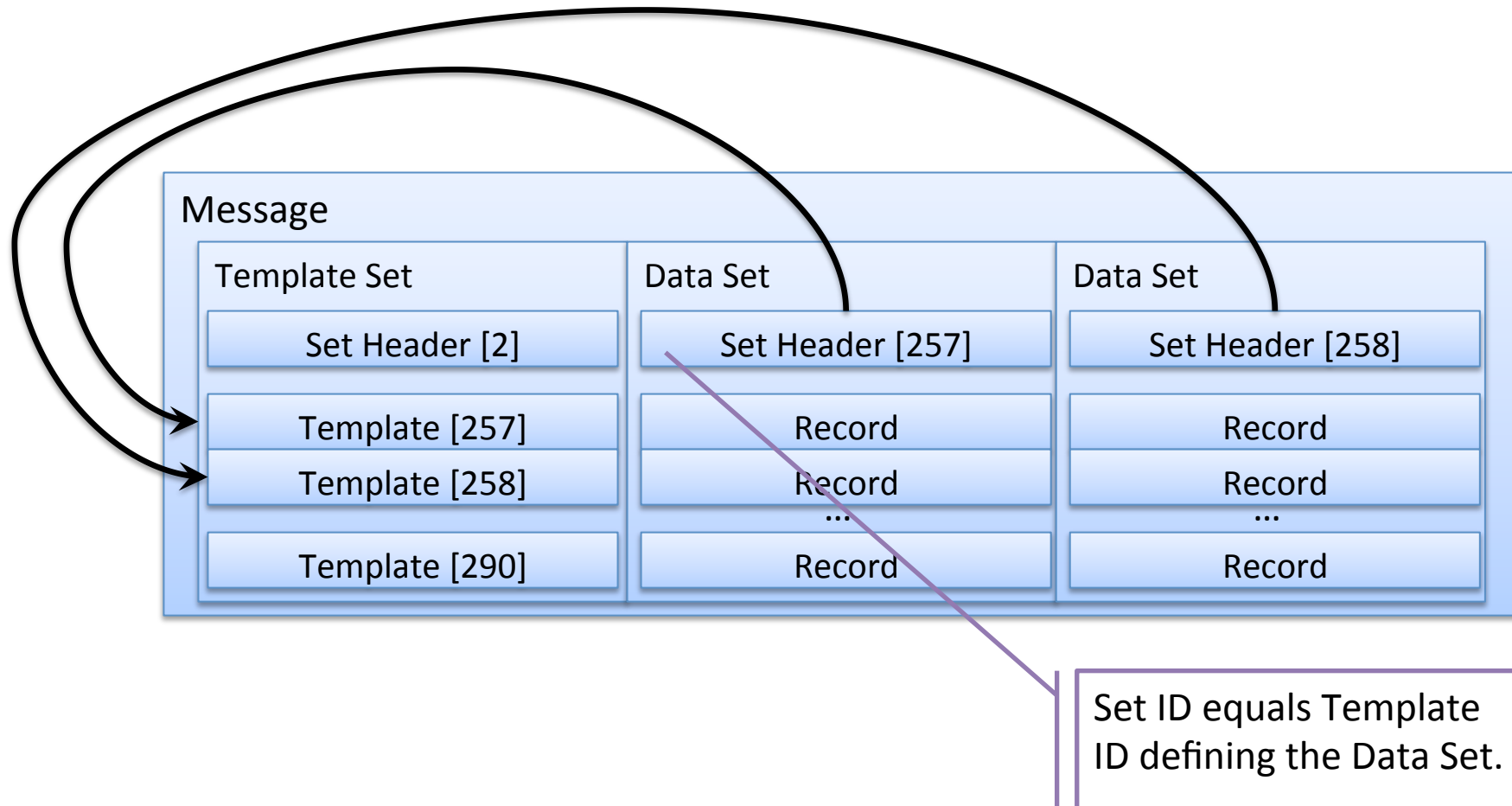
Template Structure



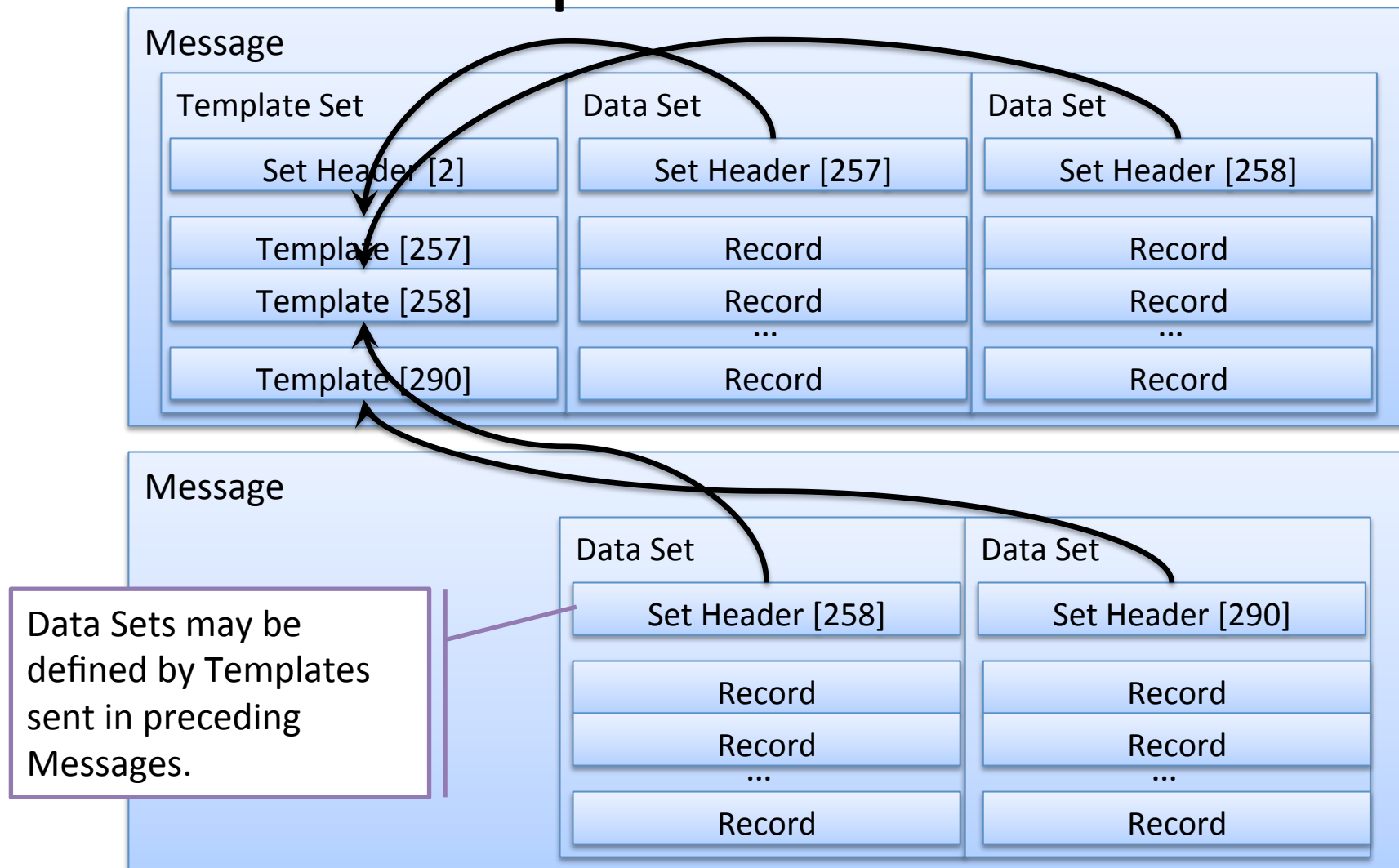
Templates and Sets



Templates and Sets



Templates and Sets



Standard Information Elements

- Information Model covers nearly all common flow collection use cases:
 - “traditional 5 tuple”:
sourceIPv4Address, destinationTransportPort, etc.
 - packet treatment:
ipNextHopIPv4Address, bgpDestinationAsNumber, etc.
 - Timestamps to nanosecond resolution:
flowStartSeconds, flowEndMilliseconds, observationTimeMicroseconds, etc.
 - IPv4, IPv6, ICMP, UDP, TCP header fields:
ipTTL, icmpTypeIPv6, tcpSequenceNumber, etc.
 - Sub-IP header fields:
sourceMacAddress, wlanSSID, mplsTopLabelStackSection, etc.
 - Various counters:
packetDeltaCount, octetTotalSumOfSquares, tcpSynTotalCount, etc.
 - Flow metadata information:
ingressInterface, egressInterface, flowDirection, ingressVRFID, selectorID, etc...
- >400 defined at <http://www.iana.org/assignments/ipfix>

Extending the Information Model

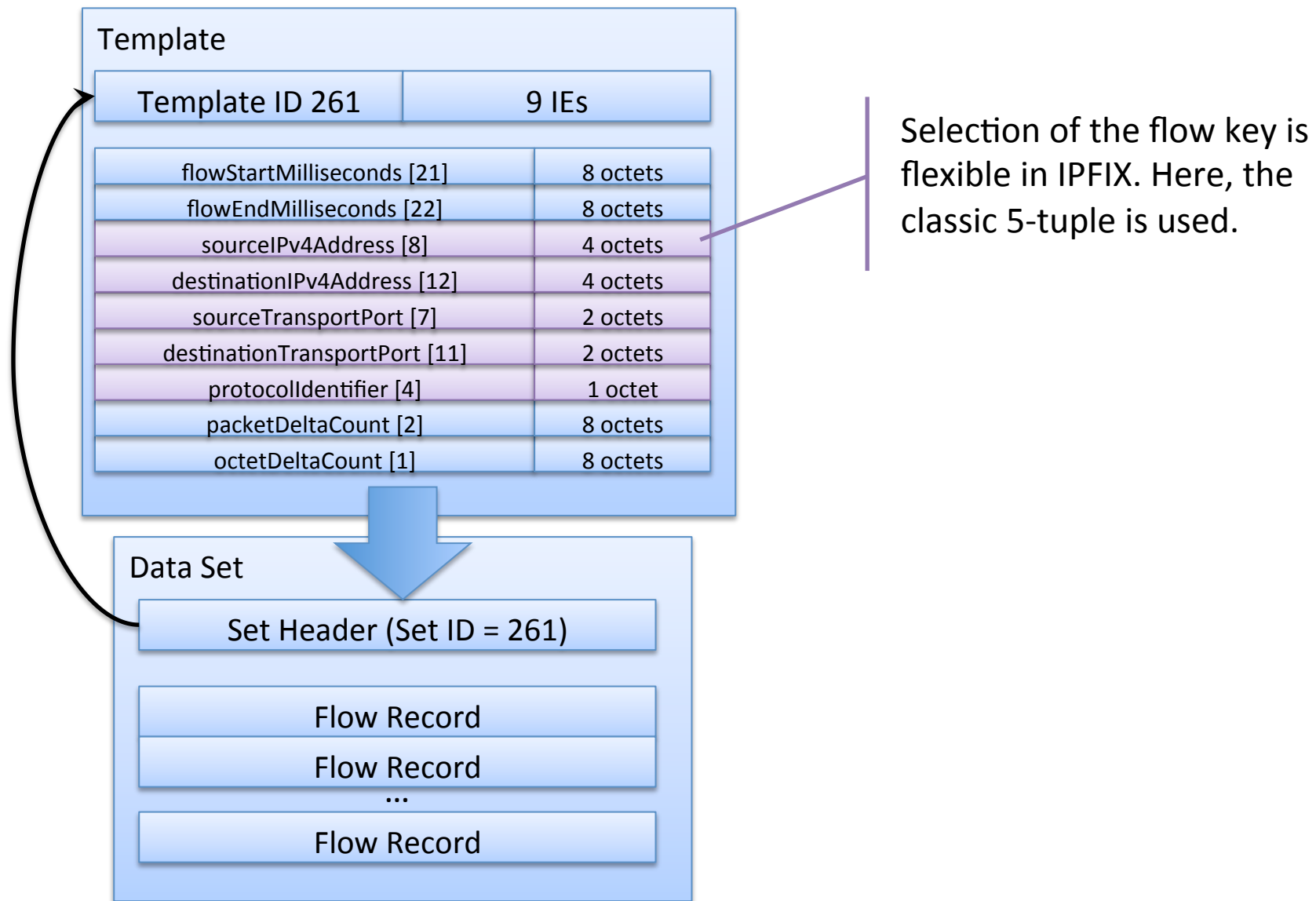
- IANA registry extensible via Expert Review, guidelines/review procedures in RFC 7013
- Experimental, commercially sensitive, or otherwise private Information Elements may be defined as *enterprise-specific*.
 - Each enterprise-specific IE number within a Template is associated with an additional private enterprise number (PEN).

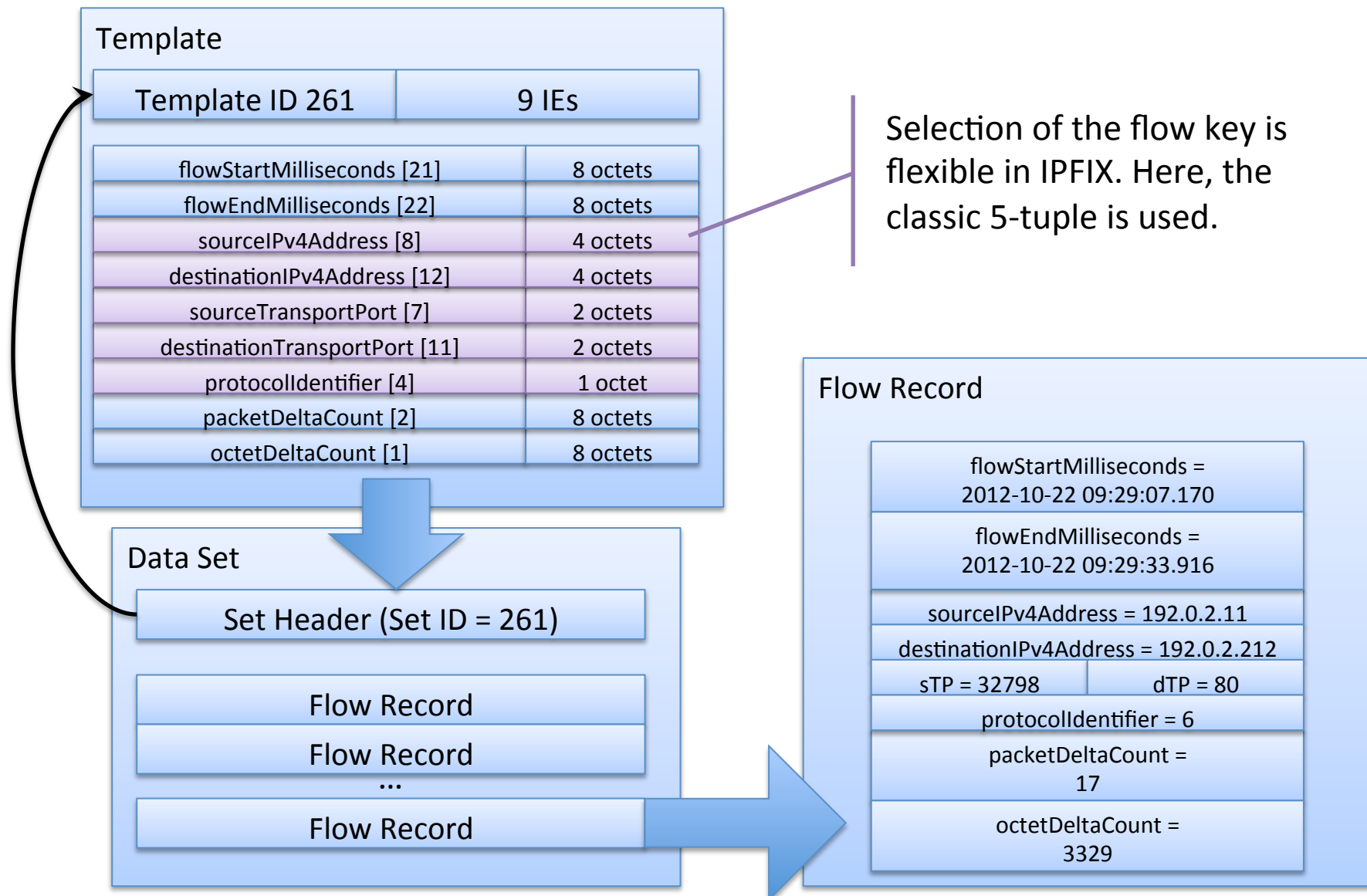
Information Element Length

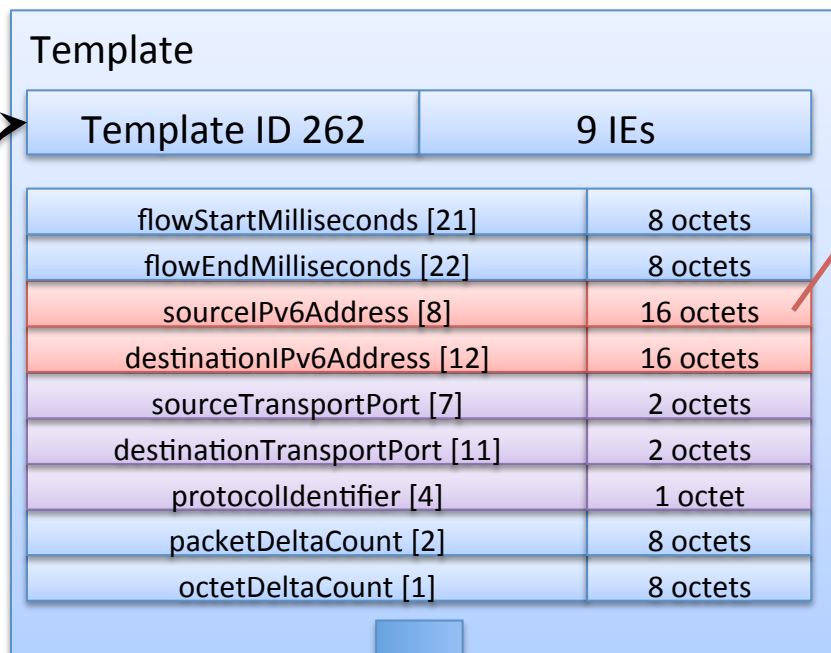
- Each Information Element has a native length associated with its data type:
 - IPv6 addresses are 16 octets, IPv4 addresses are 4 octets, and so on.
- *Reduced-length encoding* can be used to increase export efficiency.
 - e.g., a Template for use with packet and octet count that will never overflow 2^{32} can be encoded in 4 octets, instead of the native 8.
 - e.g., interface numbers: many devices can get away with 1 byte.
- *Variable-length encoding* can be used to efficiently export variable length data.
 - One-byte length-prefix up to 254 bytes (i.e., Pascal-style string)
 - Three-byte length prefix up to 65515 bytes
 - e.g. wlanSSID, which is a string.

Template	
Template ID 261	9 IEs
flowStartMilliseconds [21]	8 octets
flowEndMilliseconds [22]	8 octets
sourceIPv4Address [8]	4 octets
destinationIPv4Address [12]	4 octets
sourceTransportPort [7]	2 octets
destinationTransportPort [11]	2 octets
protocolIdentifier [4]	1 octet
packetDeltaCount [2]	8 octets
octetDeltaCount [1]	8 octets

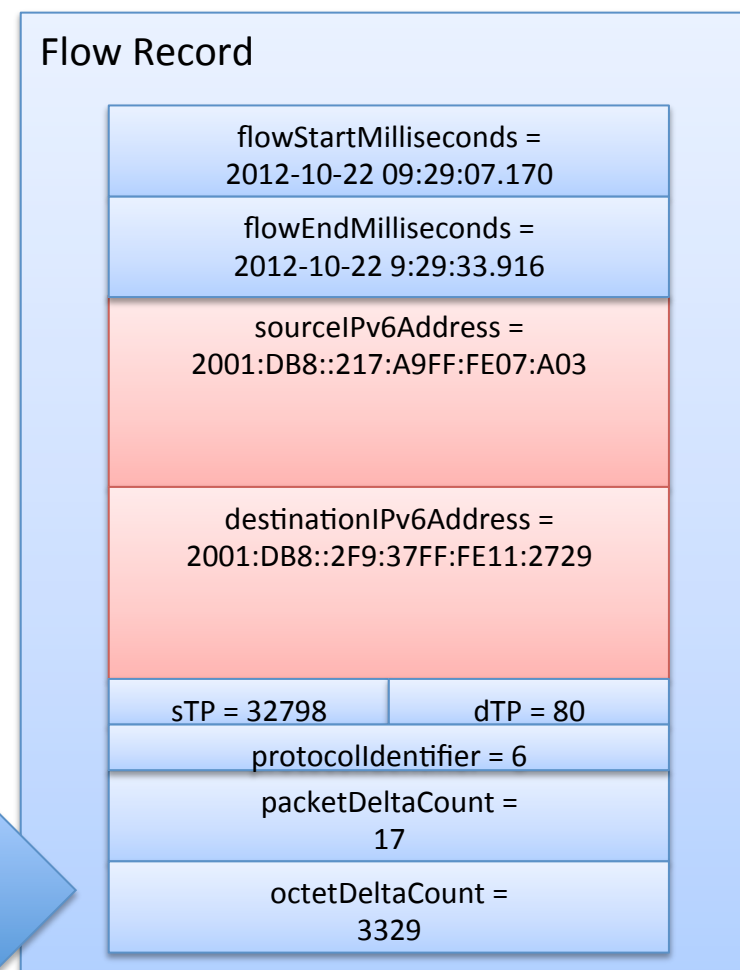
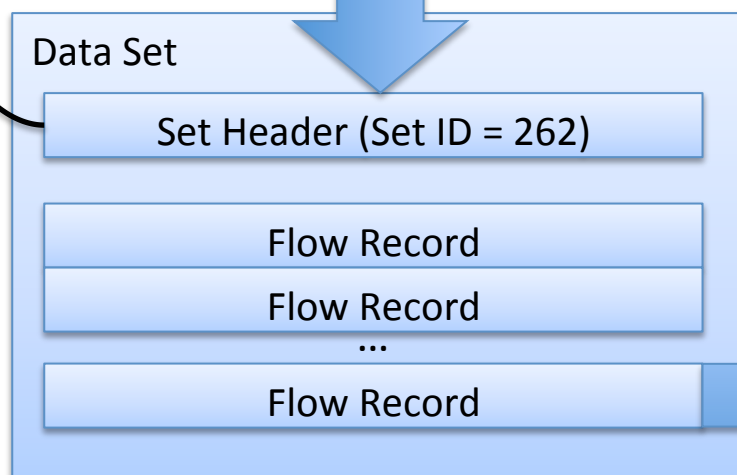
Selection of the flow key is flexible in IPFIX. Here, the classic 5-tuple is used.

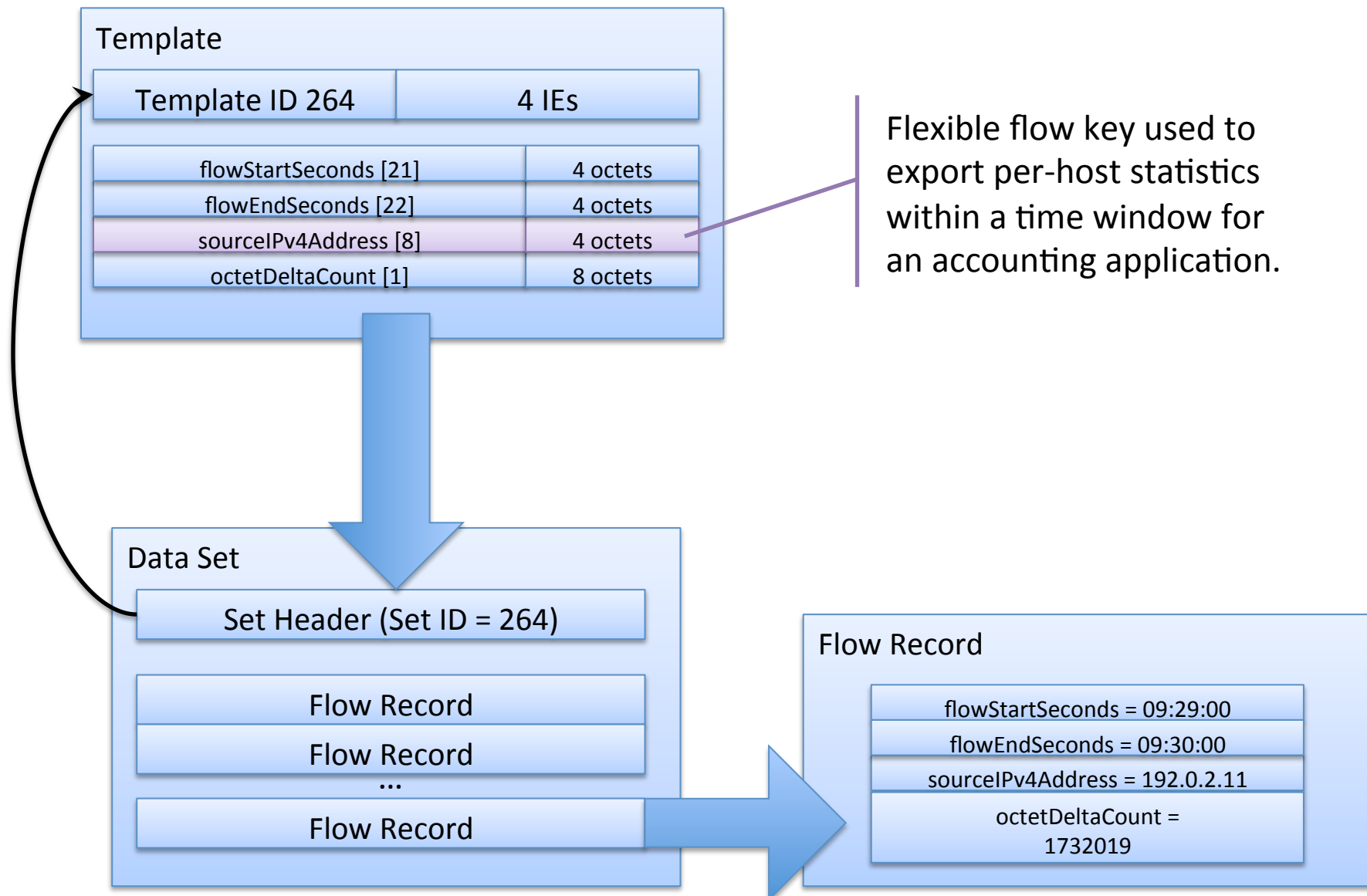






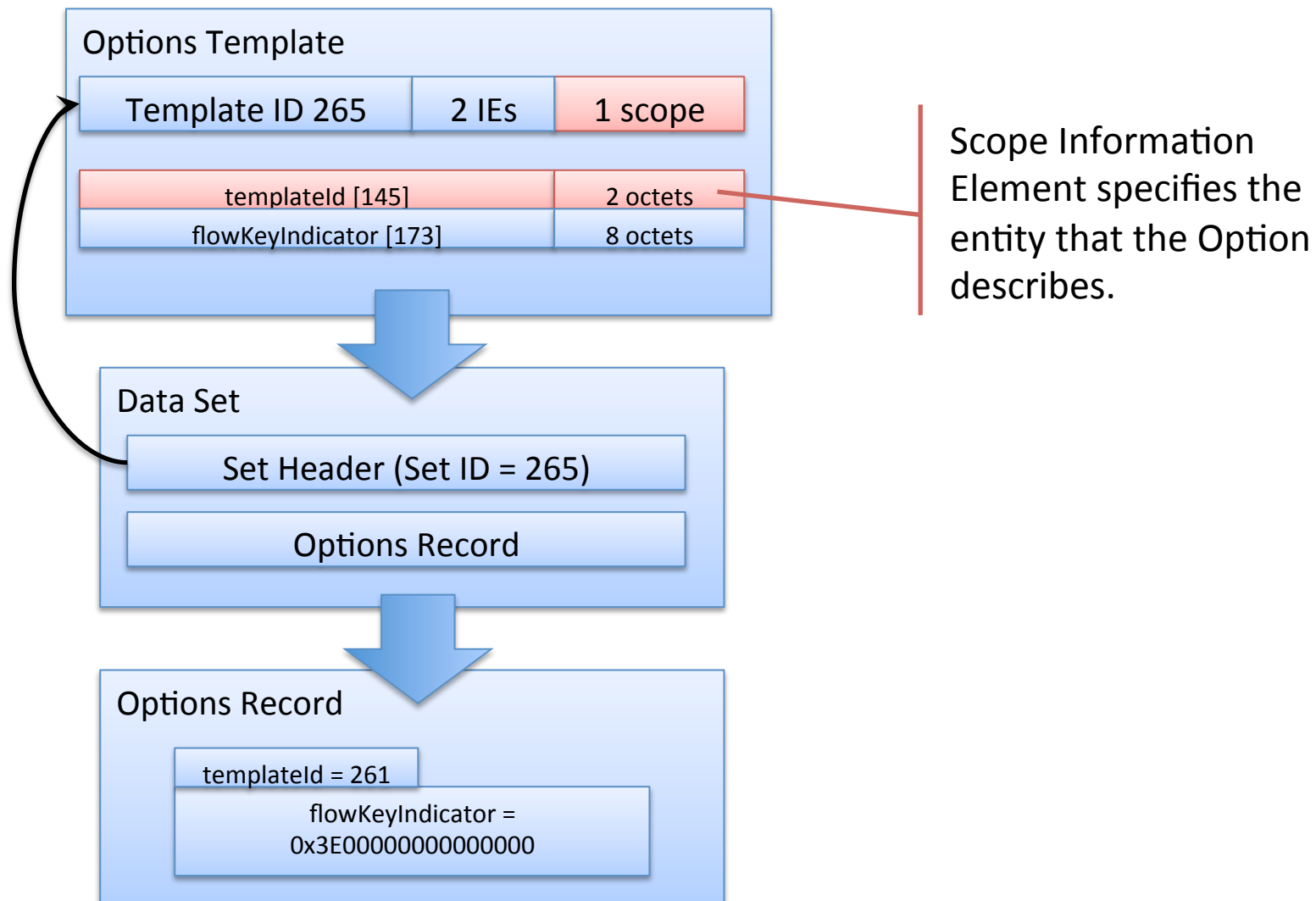
Variations on a record format defined by defining a new template with different IEs.

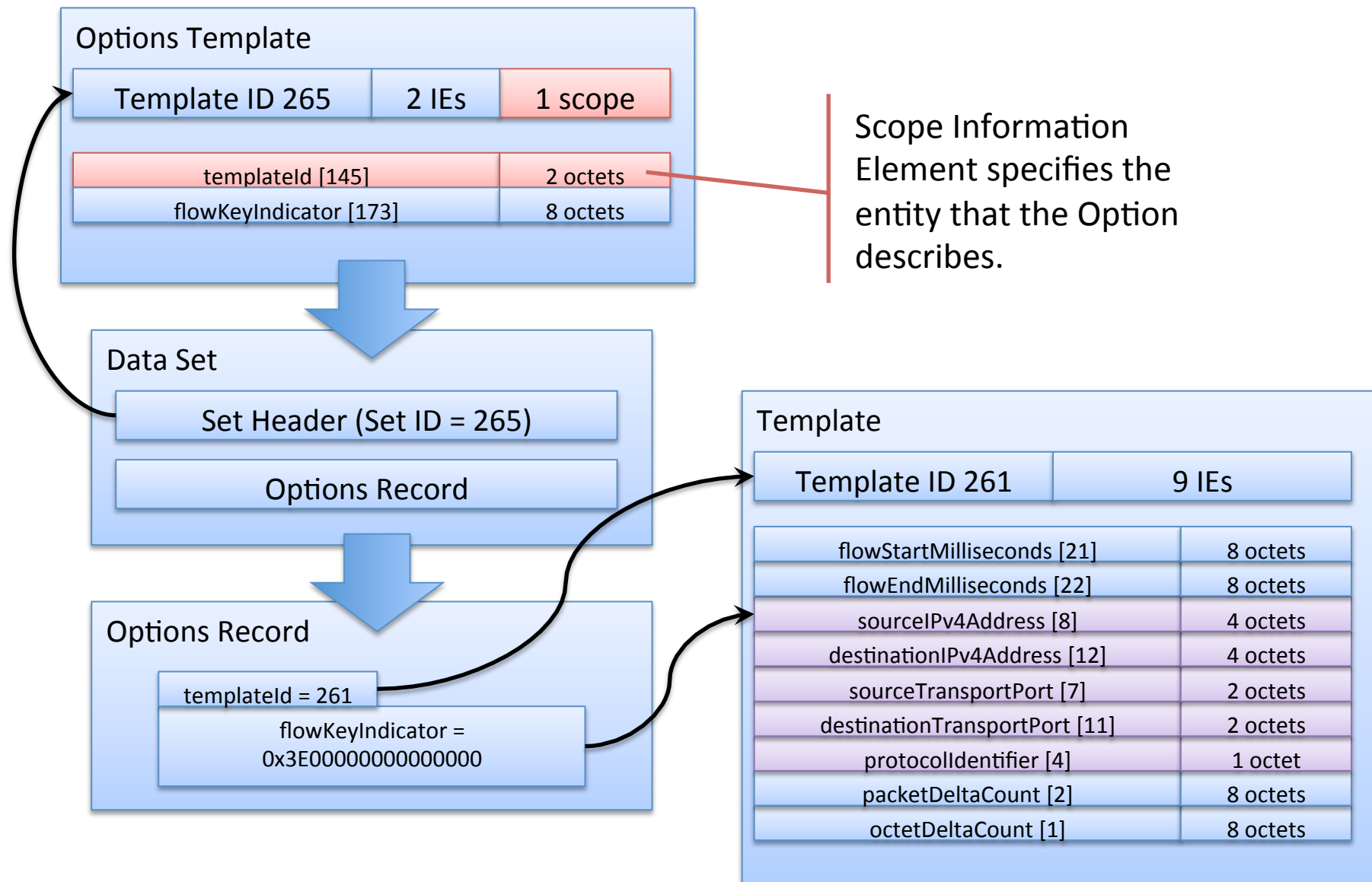




Options

- *Options Templates* are a special type of Template used to define records (*Options*) bound to a specified *scope*.
 - A scope can define an entity in the real world or the IPFIX Architecture or Protocol (e.g., an Exporting Process, a Template), or a property of some set of flows.
- While Flow Records describe Flows, Options Records describe things other than Flows:
 - information about the collection infrastructure (e.g. reliability statistics),
 - metadata about flows or a set of flows, or
 - common properties of a set of flows.





IPFIX Files (RFC 5655)

- IPFIX File: serialized stream of IPFIX Messages
 - “Filesystem” transport for IPFIX
 - Useful for storage, document-based workflow, embedding IPFIX data in named-resource-oriented protocols
- Simplicity of representation improves flexibility
 - no additional structure beyond IPFIX Messages → freely appendable, embeddable.



IPFIX Structured Data

- IPFIX supports flat data
- How do we represent...
 - A list of output interfaces in a multicast flow?
 - A list of AS in the BGP AS path?
 - A list of MPLS label stack entries?
 - A list of (time, performance metrics)?
- RFC 6313: Export of Structured Data in IPFIX

IPFIX Structured Data

- **basicList**
 - represents a list of zero or more instances of any single Information Element, primarily used for single-valued data types. For example, a list of port numbers, list of interface indexes, list of AS in a BGP AS-PATH, etc.
- **subTemplateList**
 - represents a list of zero or more instances of a structured data type, where the data type of each list element is the same and corresponds with a single Template Record. For example, a structured data type composed of multiple pairs of ("MPLS label stack entry position", "MPLS label stack value"), a structured data type composed of performance metrics, a structured data type composed of multiple pairs of IP address, etc.
- **subTemplateMultiList**
 - represents a list of zero or more instances of a structured data type, where the data type of each list element can be different and corresponds with different template definitions. For example, a structured data type composed of multiple access-list entries, where entries can be composed of different criteria types

Example: Performance Metrics

- Mediation: Data aggregation, reduction, correlation, and analysis
 - Aggregation in space (different line cards in the router)
 - Aggregation in time (performance metrics)
- Simple equation:
 - IPFIX in branch office
 - + WAN export bandwidth limitation
 - + performance metrics sent on regular basis for performance assurance
 - + IPFIX export from different observation domains in the router
 - = mediation function + IPFIX structured data

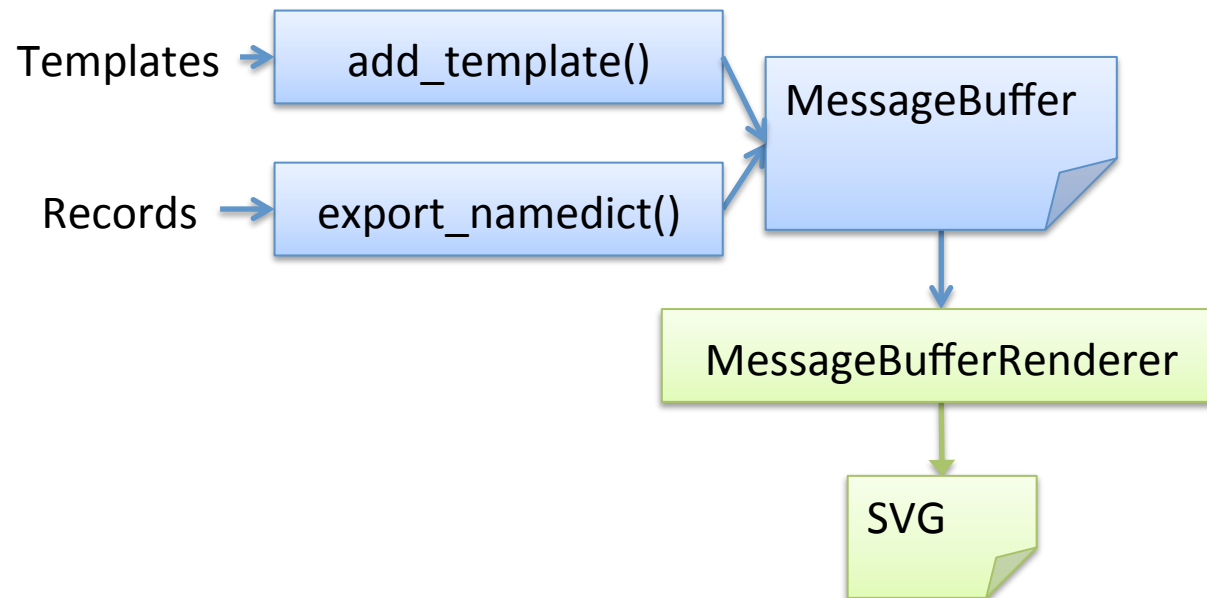
Some running code to go with the rough consensus

DEMONSTRATION

Demonstration: Exploring Structure

- ipfix module for Python 3.3+
 - <http://pypi.python.org/pypi/ipfix>
 - <http://britram.github.io/python-ipfix>
 - `pip install ipfix`
- We'll build messages with Python API and render them to SVG diagrams
 - undocumented `ipfix.vis` module
- Show what IPFIX looks like on the wire

Demonstration: Structure

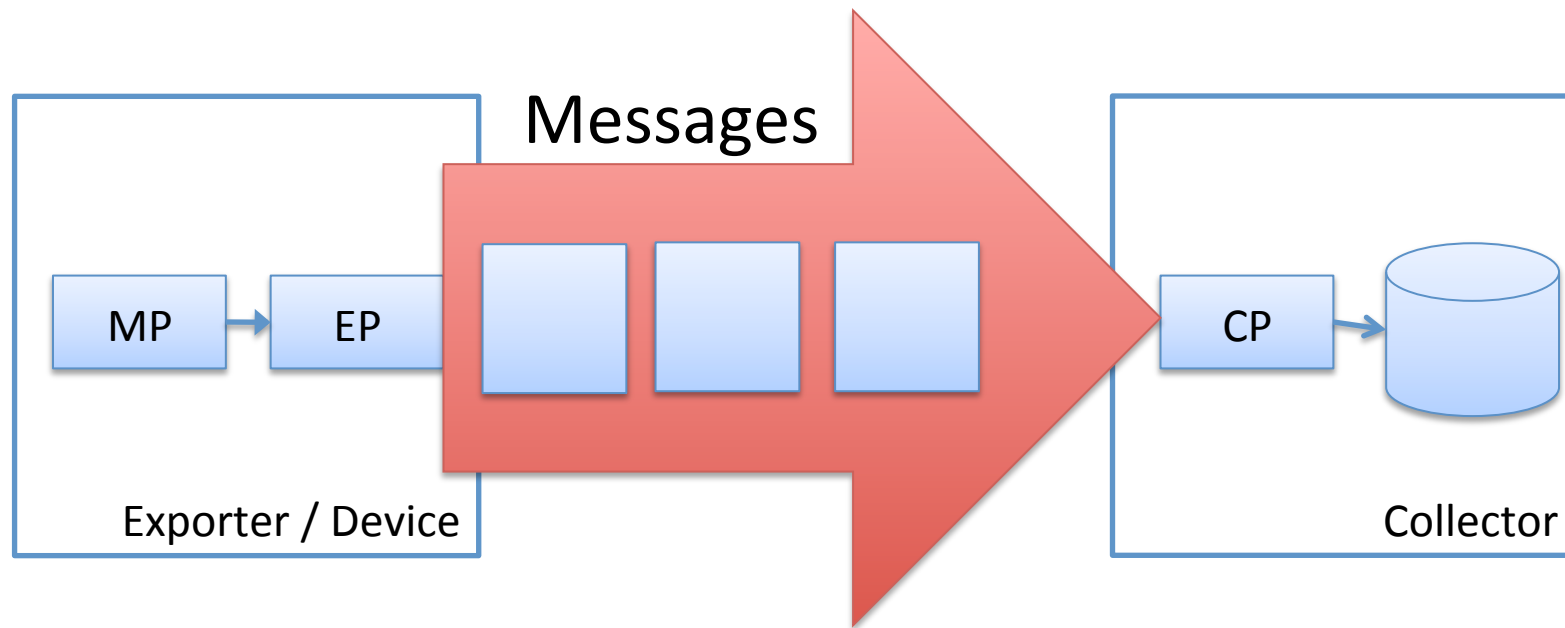


IPFIX on the Wire: Transports, Template Management, and Security

PROTOCOL DYNAMICS

IPFIX as a Message Stream

- IPFIX is a protocol for transmitting Messages from the EP to CP.
 - Unidirectional: EP initiates connection.
- IPFIX Messages map to Messages/Segments/Datagrams in underlying transport.



Transport Protocols

- SCTP
 - Mandatory to implement
 - Provides partial reliability, multiple streams
 - Some issues with implementation
- TCP
 - Intended for transport of IPFIX across the Internet
 - or implementations on devices which do not support SCTP where security (via TLS) is important.
- UDP
 - No reliability or congestion awareness
 - Intended for deployment only on devices without SCTP support, and
 - only on dedicated networks within a single administrative domain
 - i.e., as a migration path for replacement of legacy collection infrastructures.

An Introduction to SCTP

- SCTP (Stream Control Transmission Protocol) provides a sequential packet transport service.
- Supports several features beyond TCP/UDP:
 - Streams
 - Partial reliability (with PR-SCTP extension)
 - Unordered delivery
 - Transport-layer multihoming
 - Applicable mainly to mobile networks.
- Simpler state machine than TCP, with a la carte selection of features.

SCTP Partial Reliability

- PR-SCTP provides per-packet specification of reliability:
 - Reliable transport with a mechanism to skip retransmissions for certain packets.
- Allows multiple applications with different reliability requirements to run on the same association.
- Allows UDP-level best-effort reliability while still providing TCP-level congestion control.
 - Templates **MUST** be sent reliably.

SCTP Streams

- Multiple independent sequences of packets within the same association.
 - May be used to logically separate different planes (control, data) or applications.
 - Improves end-to-end delay by avoiding head-of-line blocking.
- No restrictions on the use of streams within the IPFIX protocol.
 - ...up to the limits of the underlying SCTP stack.
 - RFC 6526 provides additional features (per-template drop counting with partial reliability, fast template reuse) given some restrictions on stream usage.

Template Management

- In the simplest case, an EP defines and exports all the Templates to be used within a session at the beginning of the session.
- Templates may be withdrawn and Template IDs reused, with restrictions on operation ordering.
 - Template withdrawal: empty Template for an ID

Template Management: UDP

- IPFIX requires reliable transport for Templates.
 - UDP doesn't provide reliable transport.
- Templates are scoped to the session lifetime
 - No real definition of a session for UDP, either.
- So, template management under UDP is completely different:
 - EP resends every Template in active use periodically.
 - CP discards Templates periodically.
 - EP and CP have independently-configured retransmission and discard delays.

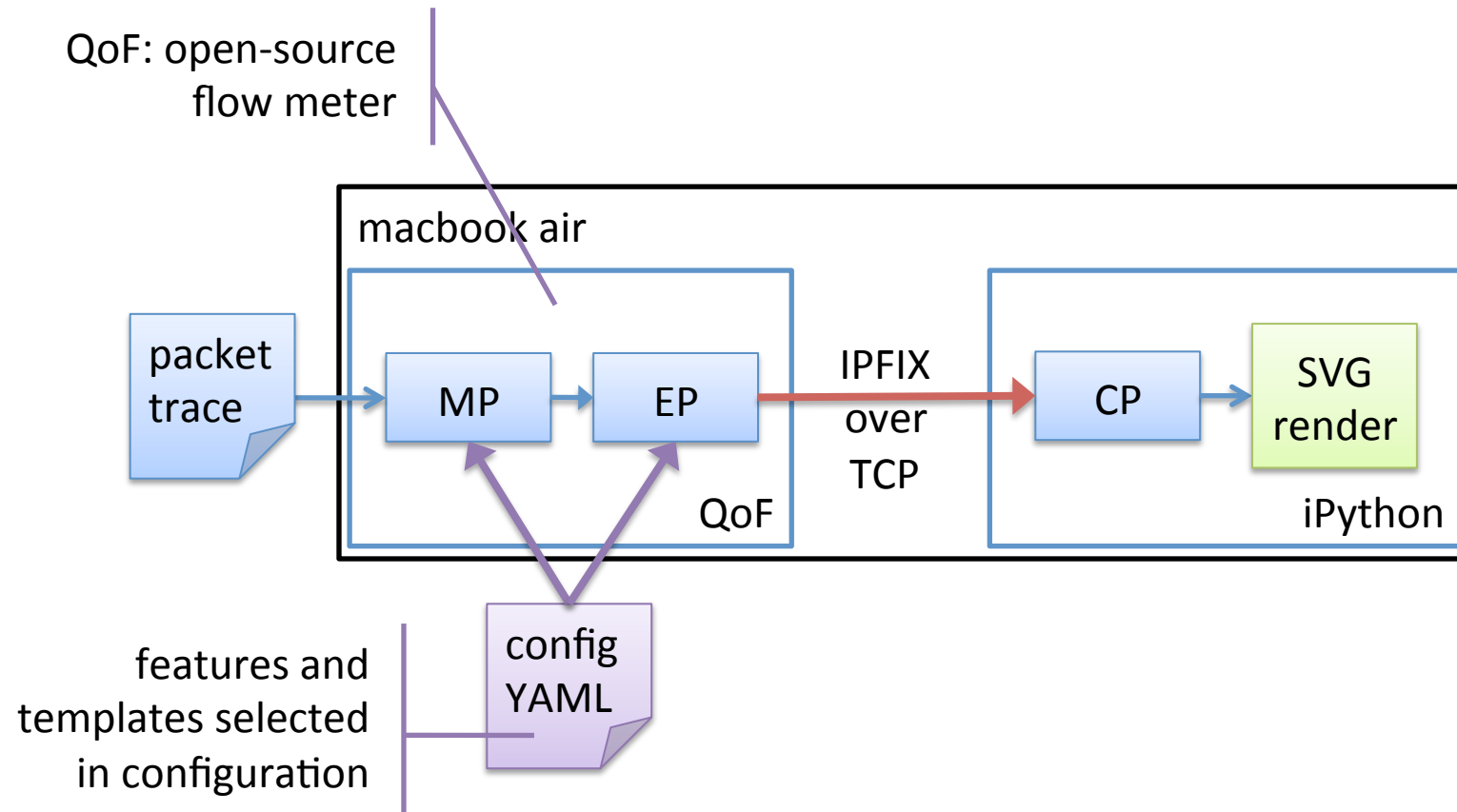
IPFIX Security

- TLS/DTLS used to secure IPFIX across uncontrolled or non-dedicated networks
 - DTLS used to secure SCTP and UDP transport.
 - TLS used to secure TCP transport.
 - Alternative: run inside a dedicated secure tunnel.
- Since the EP initiates the connection to the CP, EP acts as (D)TLS client, CP as (D)TLS server.
- IPFIX requires strong mutual authentication via X.509 certificates.

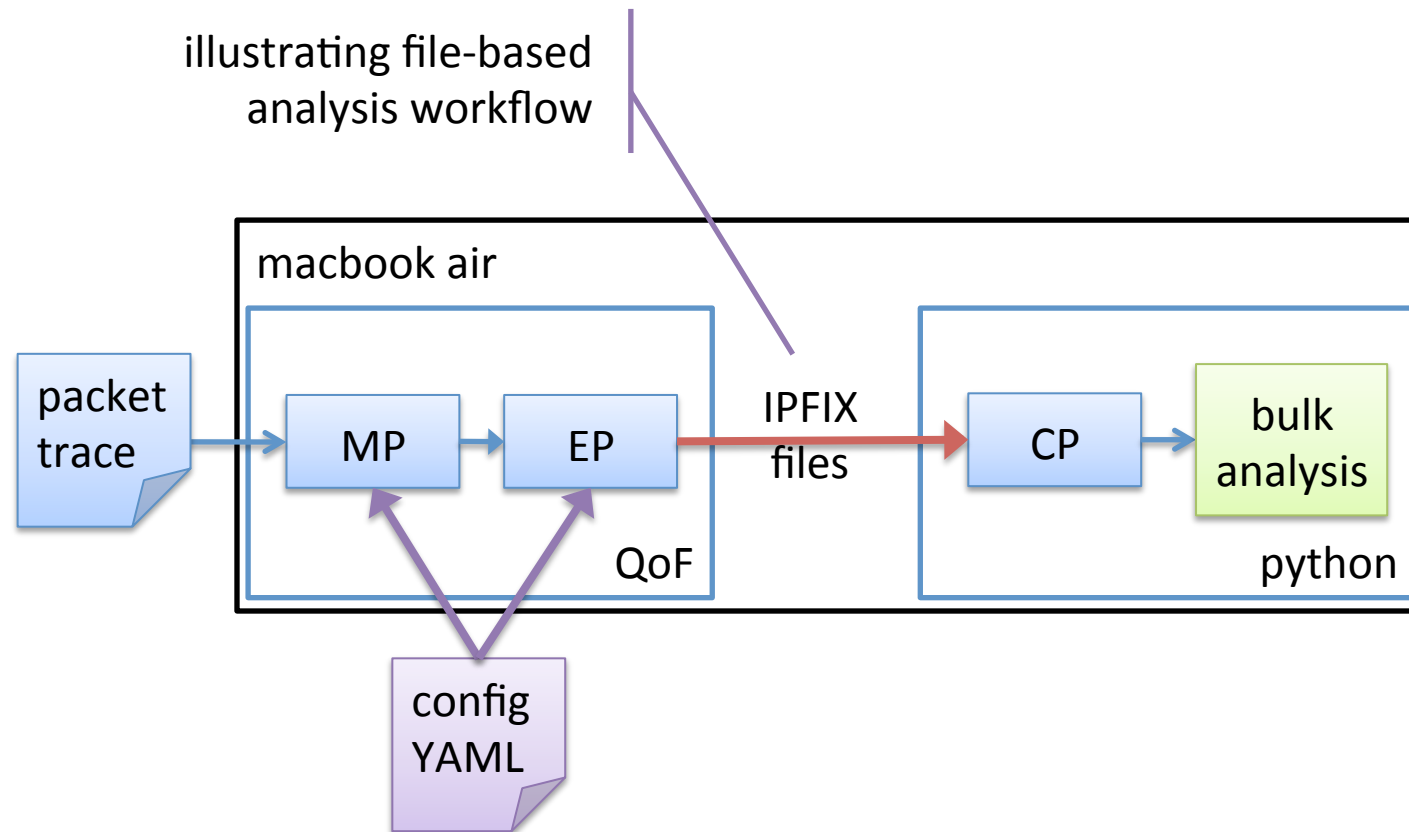
Let's have a look at some actual flows

DEMONSTRATION

Demonstration: QoF



Demonstration: QoF



Not just NetFlow Version 10

CROSS-AREA APPLICATIONS

Beyond *flow* information export

- IPFIX originally specified for flow export
 - Flow-oriented Information Model
 - Flow-specific terminology
- Applicable to any network management area requiring:
 - Unidirectional export of large number of identically structured records.
 - Self-description of record formats for flexibility
- Most application areas just need new IEs
 - <http://www.iana.org/assignments/ipfix>
 - Enterprise-specific IEs for proprietary/experimental use

Beyond flow information export:

Examples

- PSK Automatic Propagation Reporter
 - Phase Shift Keying, in the context of amateur radio
 - <http://pskreporter.info/pskmap.html>
- Use case: syslog replacement in a firewall
 - 10-Gbps flows, 100-k connections per second = lots of logs
 - Gain in terms of connection/s and throughput
- Plixer IPFIXify: unify event logs in various formats into IPFIX for transport/analysis

PSAMP (Concluded WG)

- PSAMP (Packet SAMPLing) was an effort to:
 - Specify a set of selection operations by which packets are sampled, and describe protocols by which information on sampled packets is reported to applications
- PSAMP protocol specifications
 - Agreed to use IPFIX for export protocol
- Information model for packet sampling export
 - Extension of the IPFIX information model

PSAMP (Concluded WG)

- PSAMP is the IPFIX Metering Process, with flow composed of a single packet
- Framework, RFC 5474
- Sampling and Filtering Techniques, RFC 5475
- PSAMP Protocol Specifications, RFC 5476
- PSAMP Information Model, RFC 5477

IPFIX versus PSAMP

Metering Process

This is PSAMP

(filtering, sampling, hashing)

Exporting Process

This is IPFIX

Flexible NetFlow

RFC3917: Requirements

Shared Information Model

One flow record composed of one packet?

IPFIX/PSAMP configuration via XML

IPFIX and PSAMP are complimentary: no more boundary

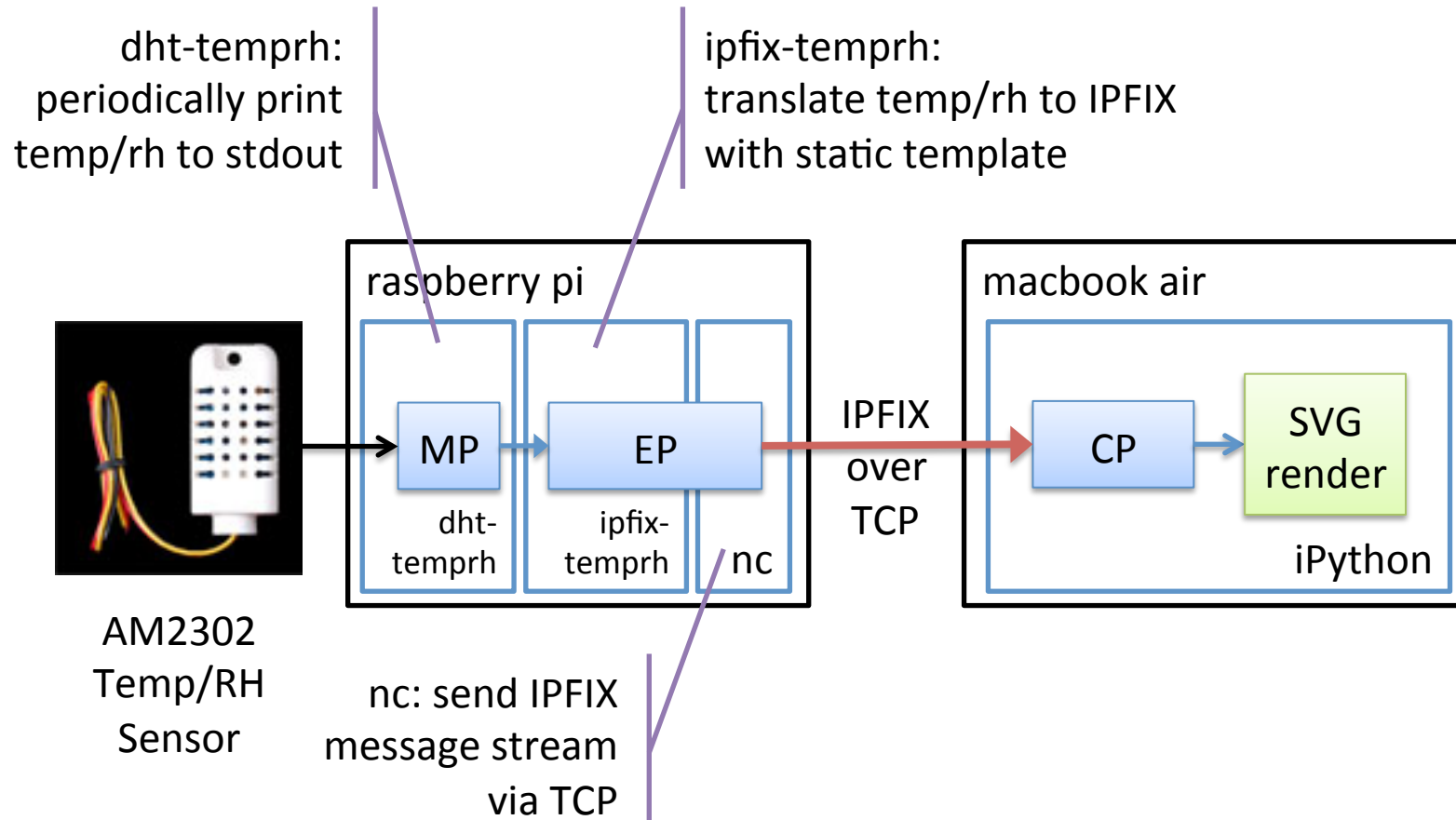
Introducing the IE-DOCTORS

- Additions to the IANA Information Element (IE) registry on Expert Review basis.
- Guidelines for experts given in RFC 7013:
 - Goal: consistency and usability
 - “New IEs should look like current IEs”
 - Reviews of IEs discussed among IE-DOCTORS, who also assist with suggested changes to IE definitions.
- Accelerated review allows many new applications to be brought to IPFIX without requiring a specification
 - ...and should allow future IPFIX extension to be done in WGs competent for that extension area, not the IPFIX WG

IPFIX export fits anywhere you have a working network stack

DEMONSTRATION

Demonstration: Beyond Flow



From the lab into the rack

APPLYING IPFIX IN OPERATIONS

Open-source IPFIX meters/exporters

- QoF (<https://github.com/britram/qof>)
 - Focus on TCP performance measurement, analysis of payload-free traces, flexible output templates
 - GPL, fork of...
- YAF (<http://tools.netsa.cert.org/yaf>)
 - Focus on network security and traffic classification with DPI features
- nProbe (<http://ntop.org/>)
- Open vSwitch (<http://openvswitch.org/>) (PSAMP)

Selected vendors shipping IPFIX export*

- Avaya Networks
- Barracuda Networks
- Cisco Networks
 - NGA 3240
- Citrix (NetScaler)
- Dell (SonicWALL)
- Extreme Networks
- F5 networks
- Gigamon
- Juniper Networks
 - MX240/480/960, 10.2
- Nortel Networks
 - ERS 5500/8600
- Ubiquiti Networks
- VMWare (ESX vswitch)
- Xirrus
- Thanks to Andrew Feren at Plixer, see <http://www.plixer.com/Scrutinizer-Netflow-Sflow/configuring-netflow-ipfix-sflow.html>

Selected IPFIX Collection and Analysis Tools

- nTop (<http://ntop.org/>), open source
- SiLK (<http://tools.netsa.cert.org/silk>)
 - GPL, UNIX-like CLI tool chain for flow analysis
- Plixer Scrutinizer <http://plixer.com/>, commercial
- Many other NetFlow V5/V9 analyzer vendors with varying levels of support for IPFIX

FIN

- IPFIX provides an information model, self-describing data format, and transport protocol for representing and transferring network event information.
- Focused on flow measurement, applicable to most network management activities.
- Questions? Now, or later:
trammell@tik.ee.ethz.ch