IPFIX Mediation: Framework

<draft-kobayashi-ipfix-mediator-model-01.txt>

Atsushi Kobayashi, Keisuke Ishibashi,
Kondoh Tsuyoshi(NTT)
Daisuke Matsubara(Hitachi)

Update IPFIX Mediator draft

- Mainly, this draft updates as follows.
  - Refines several sentences based on several comments.
    - Paul's comments are valuable. Thanks!
  - Towards writing up of framework draft, I refined terminology Section.

What is IPFIX Mediator?

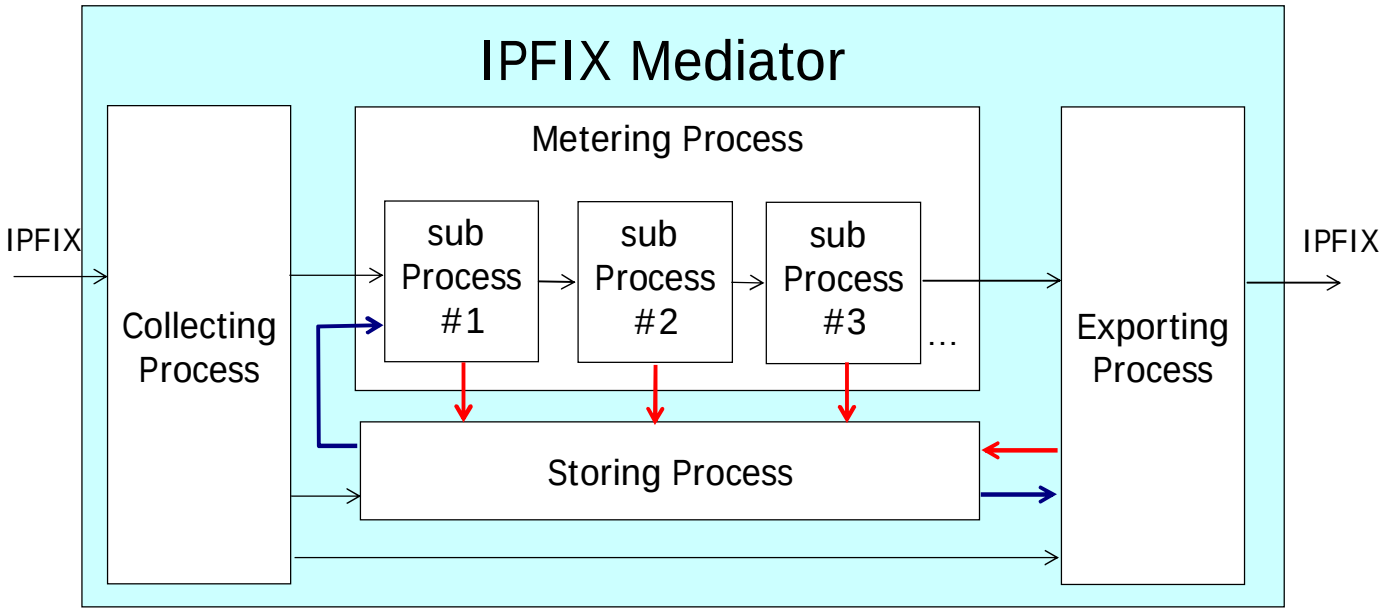
- To clarify, I listed up 2 type of IPFIX Mediator.
  - IPFIX Protocol Mediation
    - relaying and multiplexing of IPFIX Transport Sessions
    - Not change the set of Flow Records nor the value or Template of Flow Records.
  - Flow Mediation
    - Creates a new set of Flow Records from input Flow Records.
      - Aggregation
      - Selection
      - Modification
        - Changing the value of specified Information Elements.
        - Changing the Template by adding/deleting specified Information Elements.

IPFIX Mediation device

- Added IPFIX Distributor device as IPFIX Mediators.
  - IPFIX Proxy
  - IPFIX Distributor
    - It replicates Flow Records and forwards them to multiple Collectors.
    - In addition, it classifies Flow Records to work as Collector load-balancer.
  - IPFIX Concentrator
  - IPFIX Firewall

Internal Component Model

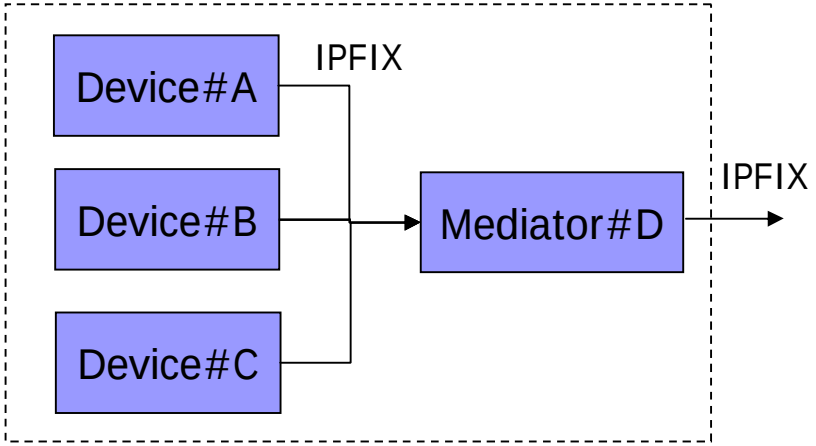
- Makes any Metering sub-process and Exporting Process able to send Data to the Storing Process.
  - Any Flow Records can be stored.
- To retrieve Flow Records, the route from Storing Process to Metering/Exporting Process was added.



Sub-Processes: Selecting Process, Aggregating Process, and Modifying Process

Observation Domain ID Management

- ODID indicates the largest set of Observation Points in original Exporters.
  - ODID is RECOMMENDED to be assigned a unique value per IPFIX Mediator.



Mediator#D reports a new ODID. It indicates the largest set of Observation Points.

- Note: IPFIX-protocol draft says that ODID should be 0, if MP aggregates input Flow Records.

Next step

- By extracting section 2 and 3.1, I will write up first framework draft.

Table of Contents

1. Introduction
2. Terminology
3. Framework for IPFIX Mediators
    1. Internal Components Model
        1. Collecting Process
        2. Metering Process
        3. Exporting Process
        4. Storing Process
    2. IPFIX Protocol Considerations
        1. Export Time Issue
        2. Observation Domain ID Management
        3. Template Management
        4. Transport Session Management
        5. Option Template Management
        6. Reporting of Exporter Information
4. Solution Scenarios with IPFIX Mediators
5. Mediator Option Template Presentation
    1. Exporter Information Option Template
    2. Usage of Scope Field
6. Security Considerations

Next step

- Section 3.2 and section 5 "IPFIX Protocol Considerations" part could be described as "IPFIX Protocol Mediation Common Parts" draft.
  - Aggregation and anonymization draft refer to it.

Table of Contents

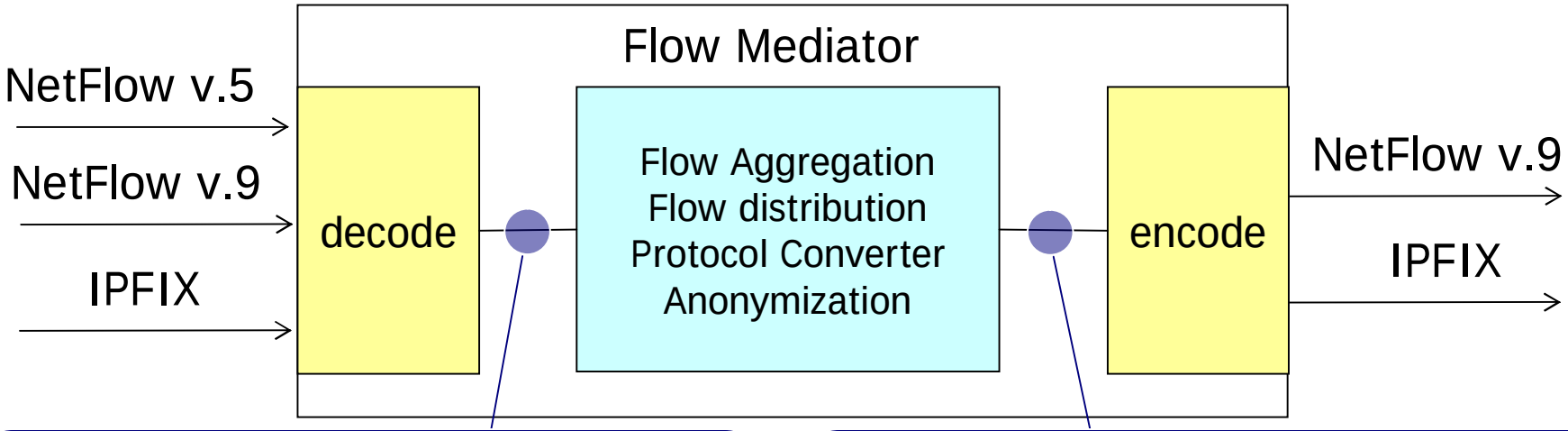
1. Introduction
2. Terminology
3. Framework for IPFIX Mediators
   1. Internal Components Model
      1. Collecting Process
      2. Metering Process
      3. Exporting Process
      4. Storing Process
   2. IPFIX Protocol Considerations
      1. Export Time Issue
      2. Observation Domain ID Management
      3. Template Management
      4. Transport Session Management
      5. Option Template Management
      6. Reporting of Exporter Information
4. Solution Scenarios with IPFIX Mediators
5. Mediator Option Template Presentation
   1. Exporter Information Option Template
   2. Usage of Scope Field
6. Security Considerations

Next step

- Implementation of IPFIX Mediator
  - Input/output Flow Record parts are already available from CPAN.
    - You can write and try your own Mediator/Exporter/Collector ideas easily.
  - Then, we can discuss about framework more clearly.

You can feel Flow Mediation

- Net::Flow perl module is available on CPAN.
  - http://search.cpan.org/~akoba/Net-Flow-0.02/
  - It can decode and encode NetFlow/IPFIX packets.
  - The decoding and encoding functions are similar IF.




```
my ( $HeaderHashRef,
     $TemplateArrayRef,
     $FlowArrayRef,
     $ErrorsArrayRef) =
     Net::Flow::decode(
                    ¥$packet,
                    $TemplateArrayRef  ) ;
```

```
my ( $EncodeHeaderHashRef,
     $PktsArrayRef,
     $ErrorsArrayRef) =
     Net::Flow::encode(
                    $EncodeHeaderHashRef,
                    ¥@MyTemplates,
                    $FlowArrayRef,
                    1400 ) ;
```