Junos inline-monitoring (IMON)

David Roy – Orange AS 3215

Feb. 2021

Who am I?

- Senior Network Support Engineer at Orange (AS 3215 – French Domestic and Mobile Backbone)
- Work with big routers: especially Juniper & Nokia SROS devices
- Passionate by networking (design/testing/troubleshooting) and for some time now by development (Python & I'm currently learning Go)
- Wrote several Juniper Day One Books and The MX Series 2nd Edition Book
- JNCIEx3
- Twitter: @door7302
- … and Sponge Bob fan ☺

#1 What?

What is Inline Monitoring aka. IMON?

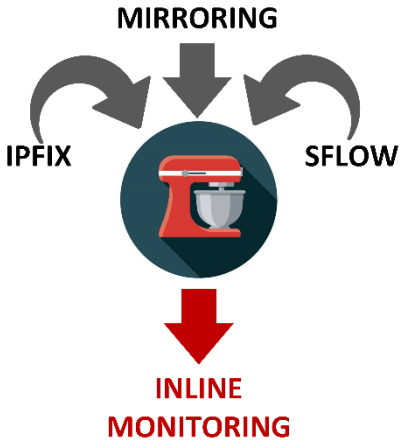
What is Inline Monitoring aka. IMON?

Yet Another Next Generation sampling solution?

- **No. It's not really the sampling YANG ☺ but …**

This feature tries to leverage the best things of other "sampling" technologies to build a flexible and scalable monitoring solution

Offload complex post processing tasks on collector side



Current features list

- Supported on all MX (including MX10k) Platforms / Linecards (ZT ASIC supported since 20.4) and vMX
- Packet sampling: capture packets from the Ethernet Header: 64B up to 126B (configurable)
- Add metadata: Input / Output Interface Index / Direction / Original packet size.
- Use IPFIX “container” + leverages IPFIX templating (easy to add metadata in the future)
- No latency - No flow aggregation/buffering
- No MPC/PFE adherence (unlike classical IPFIX)
- Several instances (up to 16)
- Several collectors per instance (up to 4 per instance)
- Per collector sampling rate
- Flexible: applied through Firewall Filter
- Currently supported for inet and inet6 families
- Applicable on Ingress and/or Egress directions
- Configurable DSCP/Forwarding Class/Routing Instance for inline monitoring packets
- Can be combined in parallel with other sampling solutions such as classical IPFIX flow aggregation.

Let’s compare the several sampling technologies available on Junos

(*some features are HW dependent*)

Sampling solutions on Junos

	Classical Mirroring	sFLOW	IPFIX	INLINE MONITORING (IMON)
Min/Max sampling rate	1:1 to 1:65535	1:1 to 1:65535	1:1 to 1:16M	1:1 to 1:16M
Filtering Level	Firewall Filter granularity	Per Logical Interface	Firewall Filter granularity	Firewall Filter granularity
Latency	usec	msec	sec to minutes (due to flow aggregation)	usec
Implementation	Inline ASIC (PFE)	CPU of LineCard (LC)	Inline ASIC (PFE)	Inline ASIC (PFE)
Limitation	N/A (Inline)	1:9K max per LC	12M Flows per PFE – Export Rate 400K flows/sec per PFE	N/A (Inline)
Metadata	NO	YES	YES (many)	YES
Report Payload	YES	YES First 128B	NO	YES (64B up to 126B)
Transport	Several (Local, GRE, L2VPN…)	sFLOW (RFC 3176)	IPFIX (RFC 7011;7702)	IPFIX (via *dataLinkFrameSection* RFC 7133)
Min Release	Available for a long time ☺	Available for a long time ☺	Available for a long time ☺	19.4 + 20.4(for ZT based Linecard)

#2 Why?

Which use cases for Inline Monitoring?

- **Use Case 1: Traffic monitoring/accounting on edge devices**
  - Use your own template on collector side to parse “packet’s headers” and some parts of the payload and retrieve only your relevant fields.
- **Use Case 2: Security**
  - Very fast DDoS / malicious Traffic detection: no flow aggregation !
  - Identify complex attacks based on signature in the payload
- **Use Case 3: TE return channel**
  - Fast telemetry channel to monitor traffic re-routing: “close loop traffic engineering”
- **Use Case 4: Payload analysis**
  - 126Bytes allow you to parse first bytes of the payload (DPI lite): Help to detect if traffic is encrypted? Detect exotic/proprietary tunnels? Dump specific packets for post analysis (i.e. DNS)? …
- **Use Case 5: Layer 2 Accounting**
  - Per MAC accounting, may be useful for IXP
- **Use Case 6: All stuff around troubleshooting**
  - Scalable/Fast/easy to configure packet mirroring solution: “Just add an troubleshooting instance/collector on demand”

What else?

#3 How?

IPFIX Template configuration

```
{master}[edit services inline-monitoring]
lab@mydevice# show
template {
    inline-template1 {
        template-refresh-rate 60;
        option-template-refresh-rate 10;
        observation-domain-id 1;
    }
}
```

Template

```
Version: 10
Length: 44
> Timestamp: Feb  3, 2021 14:51:10.000000000 Paris, Madrid
FlowSequence: 0
Observation Domain Id: 16843411
v Set 1 [id=2] (Data Template): 384
    FlowSet Id: Data Template (V10 [IPFIX]) (2)
    FlowSet Length: 28
  v Template (Id = 384, Count = 5)
      Template Id: 384
      Field Count: 5
    > Field (1/5): INPUT_SNMP
    > Field (2/5): OUTPUT_SNMP
    > Field (3/5): DIRECTION
    > Field (4/5): dataLinkFrameSize
    > Field (5/5): dataLinkFrameSection
```

metadata

See: *https://www.iana.org/assignments/ipfix/ipfix.xhtml*

Option Template

```
Version: 10
Length: 36
> Timestamp: Feb  3, 2021 14:51:30.000000000 Paris, Madrid
FlowSequence: 5071
Observation Domain Id: 16843360
v Set 1 [id=3] (Options Template): 640
    FlowSet Id: Options Template (V10 [IPFIX]) (3)
    FlowSet Length: 20
  v Options Template (Id = 640) (Scope Count = 1; Data Count = 1)
      Template Id: 640
      Total Field Count: 2
      Scope Field Count: 1
    > Field (1/1) [Scope]: FLOW_EXPORTER
    > Field (1/1): SAMPLING_INTERVAL
    Padding: 0000
```

User Value 8 bits	Reserved (0x01)	Inline Mon. Instance index 4 bits	Collector Index 3 bits	FPC slot 5 bits	Local PFE ID 4 bits

Domain ID

IMON instance / collector configuration

```
{master}[edit services inline-monitoring]
lab@mydevice# show
instance {
    myinstance_1 {
        template-name my_inline_template;
        maximum-clip-length 126;
        collector {
            collector1 {
                source-address 8[redacted].6;
                destination-address 1[redacted].3;
                destination-port 6667;
                sampling-rate 1;
            }
            collector2 {
                source-address 8[redacted].6;
                destination-address 1[redacted]01;
                destination-port 6667;
                sampling-rate 1000;
            }
        }
    }
    myinstance_2 {
        template-name my_inline_template;
        maximum-clip-length 64;
        collector {
            collector3 {
                source-address 8[redacted].6;
                destination-address 1[redacted]00;
                destination-port 6667;
                sampling-rate 8000;
            }
        }
    }
}
```

Instance Name (use later in FWF) up to 16 instances

Clip Length from 64B to 126B

Up to 4 collectors per Instance

Template reference (previous slide)

Sampling-rate per collector

More collector options:

```
{master}[edit services inline-monitoring]
lab@mydevice# set instance myinstance_1 collector collector1 ?
Possible completions:
  <[Enter]>            Execute this command
+ apply-groups         Groups from which to inherit configuration data
+ apply-groups-except  Don't inherit configuration data from these groups
  destination-address  Destination address
  destination-port     Destination port value (1..65535)
  dscp                 DSCP Value (0..63)
  forwarding-class     Forwarding class for exported frames
  routing-instance     Name of routing instance
  sampling-rate        Sampling rate (1..16000000)
  source-address       Source address
```

Cli to check statistics for instances and collectors

```
{master}
lab@mydevice> show services inline-monitoring statistics fpc-slot 9
  IMON Statistics
    FPC Slot: 9
    Packets: 246526799549, Bytes: 204462423462581
        Instance Name: myinstance
        Packets: 246526799549, Bytes: 204462423462581
            Collector Name: collector1
            Packets: 173225078057, Bytes: 143668615602612
            Collector Name: collector2
            Packets: 33464190, Bytes: 27909003873
```

Apply IMON

```
{master}[edit services inline-monitoring]
lab@mydevice# show
instance {
    myinstance_1 {
        template-name my_inline_template;
        maximum-clip-length 126;
        collector {
            collector1 {
                source-address 8[redacted]6;
                destination-address 1[redacted]3;
                destination-port 6667;
                sampling-rate 1;
            }
            collector2 {
                source-address 8[redacted]6;
                destination-address 1[redacted]1;
                destination-port 6667;
                sampling-rate 1000;
            }
        }
    }
    myinstance_2 {
        template-name my_inline_template;
        maximum-clip-length 64;
        collector {
            collector3 {
                source-address 8[redacted]6;
                destination-address 1[redacted]0;
                destination-port 6667;
                sampling-rate 8000;
            }
        }
    }
}
```

Define a FWF

```
filter IMONv4 {
    term DNS {
        from {
            protocol udp;
            port 53;
        }
        then {
            inline-monitoring-instance myinstance_1;
            accept;
        }
    }
    term CHECK_TLS {
        from {
            protocol [ tcp udp ];
            port 443;
        }
        then {
            inline-monitoring-instance myinstance_2;
            accept;
        }
    }
    term OTHER {
        then accept;
    }
}
```

!!! BEWARE !!!

Don't sample IMON traffic (self-generated) to avoid Sampling Loop !!!

Tips: define a specific term to accept, only, the udp destination-port configured for IMON.

Apply it

```
ae100 {
    unit 0 {
        family inet {
            filter {
                input IMONv4;
                output IMONv4;
            }
        }
        family inet6 {
            filter {
                input IMONv6;
                output IMONv6;
            }
        }
    }
}
```

Or

```
ae101 {
    unit 0 {
        family inet {
            filter {
                input-list [ IMONv4 ANOTHER_FILTER ];
                output-list [ IMONv4 ANOTHER_FILTER ];
            }
        }
    }
}
```

Or

```
{master}[edit forwarding-options family inet]
lab@mydevice# show
filter {
    input IMONv4;
}
```

Some info about metadata

```
  FlowSet Id: Data Template (V10 [IPFIX]) (2)
  FlowSet Length: 28
v Template (Id = 384, Count = 5)
    Template Id: 384
    Field Count: 5
  > Field (1/5): INPUT_SNMP
  > Field (2/5): OUTPUT_SNMP
  > Field (3/5): DIRECTION
  > Field (4/5): dataLinkFrameSize
  > Field (5/5): dataLinkFrameSection
```

Type of Interface	IMON on Ingress	IMON on Egress
Physical ge-* ; xe-* ; et-*	**INPUT_SNMP** = Ifindex of ingress interface	**INPUT_SNMP** = Ifindex of ingress interface
	OUTPUT_SNMP = 0	**OUTPUT_SNMP** = ifindex of egress interface
Aggregated ae*	**INPUT_SNMP** = Ifindex of ingress AE	**INPUT_SNMP** = Ifindex of ingress AE
	OUTPUT_SNMP = 0	**OUTPUT_SNMP** = ifindex of egress Child Link
IRB	**INPUT_SNMP** = Ifindex of ingress IRB	**INPUT_SNMP** = Ifindex of ingress IRB
	OUTPUT_SNMP = 0	**OUTPUT_SNMP** = ifindex of egress Vlan-Bridge encapsulated interface.

Rate of generated IMON Traffic

IMON traffic rate in Bps = (C-Rate / sampling-rate) x (74 + Clip-Size) x 8

C-Rate = Curent Frame Rate of your traffic in pps
74 = 14B (Ethernet Header) + 20B (IP Header) + 8B (UDP Header) + 32B (IPFIX IMON Header)

Ex. I've got an ae interface with a pick rate at 14Mpps and I've configured an IMON instance with these parameters:

- Sampling-rate = 8000
- Clip-Size = 126B

<u>Max. generated IMON Traffic:</u>
IMON Traffic = (14M/8000) x (74+126) x 8 = 2.8Mbps

Export example

```
  Version: 10
  Length: 158
> Timestamp: Jan 18, 2021 17:46:22.000000000 Paris, Madrid
  FlowSequence: 89875
  Observation Domain Id: 16842896
v Set 1 [id=384] (1 flows)
    FlowSet Id: (Data) (384)
    FlowSet Length: 142
    [Template Frame: 4678 (received after this frame)]
  v Flow 1
      InputInt: 527
      OutputInt: 0
      Direction: Ingress (0)
      Data Link Frame Size: 651
    v Data Link Frame Section: 4e9614752f1c00004de79d5208004500027d000000004006…
        String_len_short: 126
```

Metadata

Original packet size

You are there - IPv4

Ethernet frame

#4 Any Tests?

Tests done in lab

- **On MX2k – MPC9e (EA ASIC) based MPC (EA is the PFE of MPC 7, 8, 9 or MX204 or LC2103 MX10003 or LC2101 on MX10K….)**

- **Load a 100G interface at 96% (with a mix packet sizes): ~14Mpps**
- **2 collectors :**
  - Collector 1: sampling rate at 1:8000 – clip-length 126B
  - Collector 2: sampling rate at 1:1 – clip-length 126B
- **With classical IPFIX Flow aggregation configured in //**

- **Tested to monitor all traffic (one single term)**
- **Tested on Ingress / Egress direction (with and without filter chaining (input/output-list))**
- **Tested for inet and inet6 families**

- **Conclusion : Linerate, no traffic impact, no performance degradation, no CPU/Mem impact**
- **FYI: the collector 2 with rate 1:1 generated around 20Gbps**

Next steps (scheduled in Q1/Q2 2021)

- **Do the same tests for our legacy XL based Linecard (MPC6e)**
- **Do the same tests for our new ZT based Linecard (MPC10/11e)**
- **Apply IMON with a Forwarding Table Filter**

- **And Then …**

- **Pilot phase in live network on some edge devices**
- **Work on collector side:**
  - Stress test of pmacct/nfacctd (used during tests in lab) ???
  - Enhance goflow (https://github.com/cloudflare/goflow) to support *dataLinkFrameSection* with similar pmacct's primitive features (see next slide)??? Maybe when I will be better in Go ☺ but I'm motivated!
  - Other opensource collectors ??? Advices are welcome!

Collector side

- I used “pmacct” in lab – docker image – No stress tests have been performed on the collector.
- “Nfacctd” plugin supports IPFIX and RFC 7133. It allows you to create your own parser for decoding what you want in “dataLinkFrameSection”: “primitive” feature
- Start the collector like that:

```
# docker run -p 6667:6667/udp -v /opt/pmacct/nfacctd.conf:/etc/pmacct/nfacctd.conf -v
/opt/pmacct/primitive.lst:/var/tmp/primitive.lst -v /opt/pmacct/test.json:/var/log/test.json pmacct/nfacctd
```

Simple nfacctd.conf

```
debug: true
debug_internal_msg: false

nfacctd_port: 6667
!plugin_buffer_size: 1310720
!plugin_pipe_size: 134217728
nfacctd_disable_checks: false
!
plugins: print
!
aggregate_primitives: /var/tmp/primitive.lst
aggregate: mac_dst,mac_src,ip6_src,ip6_dst,nhead6,udp_src,udp_dst,tcp_src,tcp_dst
!
print_refresh_time: 30
print_history: 60m
print_output: json
print_output_file: /var/log/test.json
print_output_file_append: true
print_history_roundoff: m
```

Simple primitive.lst for IPv6

```
name=mac_dst packet_ptr=packet len=6 semantics=mac
name=mac_src packet_ptr=packet:+6 len=6 semantics=mac
name=ip6_src packet_ptr=l3:0x86dd+8 len=16 semantics=ip
name=ip6_dst packet_ptr=l3:0x86dd+24 len=16 semantics=ip
name=nhead6 packet_ptr=l3:0x86dd+6 len=1 semantics=u_int
name=udp_src packet_ptr=l4:17 len=2 semantics=u_int
name=udp_dst packet_ptr=l4:17+2 len=2 semantics=u_int
name=tcp_src packet_ptr=l4:6 len=2 semantics=u_int
name=tcp_dst packet_ptr=l4:6+2 len=2 semantics=u_int
```

More info:

https://github.com/pmacct/pmacct/blob/master/CONFIG-KEYS

https://github.com/pmacct/pmacct/blob/master/examples/primitives.lst.example

Collector side

```
# docker run -p 6667:6667/udp -v /opt/pmacct/nfacctd.conf:/etc/pmacct/nfacctd.conf -v
/opt/pmacct/primitive.lst:/var/tmp/primitive.lst -v /opt/pmacct/test.json:/var/log/test.json pmacct/nfacctd
```

test.json


```
{
  "event_type":"purge",
  "mac_dst":"4e:96:14:75:2f:1c",
  "mac_src":"00:00:4d:e7:9d:52",
  "ip6_src":"2[redacted],
  "ip6_dst":"2[redacted]
  "nhead6":"6",
  "udp_src":"0",
  "udp_dst":"0",
  "tcp_src":"60",
  "tcp_dst":"60",
  "stamp_inserted":"2021-01-28 17:00:00",
  "stamp_updated":"2021-01-28 17:43:54",
  "packets":2,
  "bytes":2725
}{
  "event_type":"purge",
  "mac_dst":"4e:96:14:75:2f:1c",
  "mac_src":"00:00:4d:e7:9d:52",
  "ip6_src":"2[redacted],
  "ip6_dst":"2[redacted]
  "nhead6":"6",
  "udp_src":"0",
  "udp_dst":"0",
  "tcp_src":"60",
  "tcp_dst":"60",
  "stamp_inserted":"2021-01-28 17:00:00",
  "stamp_updated":"2021-01-28 17:43:54",
  "packets":1,
  "bytes":1168
}
```

#5 Questions?

Thank you