draft-kumar-ippm-ifa-00

IETF-104
Motivation

Provide simple, protocol agnostic, flexible, inband mechanism to track and collect per flow per packet network telemetry data with low cost and high performance of HW
IFA - Recap

- Protocol agnostics proposal – Single draft handles all packet types
  - IPv4/v6 UDP/TCP, GRE, IPinIP, AH/AES, IPv6 EH, SRv6

- Same proposal works for MPLS
  - I-D. draft-Kumar-mpls-mint-00 (Kireeti Kompella, Jai Kumar, John Lemon)
  - NextHdr provides deterministic payload parsing (instead of speculative)
  - NextHdr provides ability to present other metainformation like security tag etc
IFA – Updates

– Added support for Fragmentation
  – Needed with very large number of hops in path
  – PMTU is a limitation

– Degenerate case of Fragmentation is Post Card
  – Each node generates each own metadata packet towards collector
  – No insertion of metadata in the packet

– Deployed in MSDC

– Developed and supported by System vendors
IFA – L3 Extension Header

- Use 1 experimental IP protocol type (143 – 252): PROTO_IFA
- Original IP->proto is copied in IFA->proto
- IFA header is treated as L3 upper layer protocol
- Metadata is treated as L4 information
- Metadata is treated as foreign object
  - L4 length and checksum is not updated
  - This is to ease load on hw to strip and forward
IFA – Tail Insertion

- Metadata Header is at the tail of the packet before FCS
- Metadata is inserted at the bottom of the metadata stack
- MD headers and MD stack looks inverted for packet tail stamp
IFA – Metadata Fragment (MF) Header

- IFA Header Flags “MF” bit is set by Initiator Node
- IFA Frag Header is inserted by Initiator Node
- Frag Header fields Packet ID and Fragment ID set by Initiator Node
- Collector uses timestamp, Packet ID, Frag ID and Flow tuples as a key
- Fragment ID is used to recreate ordered metadata set

IFA Header:

```
| Ver=2.0 | GNS | NextHdr=TCP/UDP | Flags | Max Length |
```

(4) Flags (8 bits)

- 0: Checksum - Indicates the presence of the checksum header.
- 1: TA - Turn Around. Indicates that the IFA packet needs to be turned around at the terminating node of the IFA zone.
- 2: I - Inband. Indicates this is live traffic. Strip and forward MUST be performed by the terminator node if this bit is set.
- 3: TS - Tail Stamp. Indicates the IFA zone is requiring tail stamping of metadata.
- 4: MF - Fragment. Indicates the presence of fragment header. ignored on receipt.
- 5-7: Reserved. MUST be initialized to 0 on transmission and ignored on receipt.

MD Frag Header:

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
| Packet ID. | FragID | L |
```
IFA – Metadata Fragmention Function

Any IFA node MAY perform MF function.

Degenerate case each node generates fragment aka IFA report instead of inserting metadata in packet

For Use When
– Very large number of hops in the network
– PMTU limits the packet size

Requirement
- “MF” bit, set by initiator node
  - 1: Fragment header present
  - 0: Do not Fragment (DF)
- IFA Frag header inserted by Initiator node
  - [Packet ID, Fragment ID]
- Nodes in path check
  
  If (ifa_hdr.flags.MF == TRUE) && (ifa_hdr.current_length > ifa_hdr.md_hdr.max_length)

  Send a *truncated copy of packet to collector
  Increment Sequence Number in IFA Frag Header
  Delete metadata stack (till the fragmentor hop)
  Insert metadata of fragmentor hop and forward packet
Frag header is a 4 byte overhead

5 bits of Frag ID will support 32 Fragmentor nodes in the path

Last Frag “L” bit indicates last fragment

Metadata Fragmentation is conditional and MAY be performed by any IFA node
  - If (MF bit is set in IFA header) and ((incoming Current length + sizeof(local metadata)) > maximum length)

Fragmentation involves following
  - Create a copy of packet and send it to collector
  - Delete the metadata stack
  - Update the current length in IFA header
  - Increment the FragID by 1 in IFA MD header
  - Terminator node SET the lastFrag bit in IFA MD header
Fragmentation Header and ID Wrap around

- Fragmentation SHOULD be enabled if exceeding PMTU or allowable Metadata stack length
- 26bit ID field MUST be unique within the MDL (maximum datagram life) and 5 tuple
- Average i-mix packet size 512 bytes with 3 sec MDL, bit rate supported for a given node before ID wraparound
  - 183.25 Gbps
  - Sampler per node MUST not exceed traffic rate of 183.25 Gbps
- Initiator node responsible for sampling need to manage the ID space
- First fragment always start with FragID “0”
- Collector should discard fragment if received after MDL
**IFA Fragmentor**

- **Initiator Node-1**
  - Flow ID: A
  - Seq No: 1

- **2 Transit Nodes 2, 3**

- **Frag on Node 4**

- **2 Transit Nodes 5, 6**

- **Frag on Node 7**

- **Transit Node 8**

- **Terminator Node 9**

- **Collector**

Legend:
- **IFA MF Header**
- **IFA Metadata Stack**
IFA Fragment Ordered Set

Collector

Key: Timestamp, Flow Tuple, PacketID
Order Key: Fragment ID

*Per Flow Packet ID can also be used to detect packet drops
Status of I-D

draft-kumar-ippm-ifa-01
   Updated with packet format interworking with tunnels and IPv6 EH
   Updated with support for fragmentation

Request
   Would appreciate review and comments from WG
   WG adoption

Reference
Thank You
BACKUP
IFA IPv4/IPv6 Encapsulation

**IPv4 Header:**
-----------------------------------
| Version | IHL | Type of Service | Total Length |
-----------------------------------
| Identification | Flags | Fragment Offset |
-----------------------------------
| Time to Live | Protocol | Header Checksum |
-----------------------------------
| Source IPv4 Address | Destination IPv4 Address |
-----------------------------------

**IFA Header:**
-----------------------------------
| Payload Length | Next |
-----------------------------------

**IPv6 Header:**
-----------------------------------
| Version | Traffic Class | Flow Label |
-----------------------------------
| Payload Length | Next Hdr = IFA | Hop Limit |
-----------------------------------
| Source IPv6 Address | Destination IPv6 Address |
-----------------------------------

**UDP (Outer VxLAN UDP, IPinIP, GRE Header etc):**
-----------------------------------
| Source Port | Dest Port = VXLAN Port |
-----------------------------------
| UDP Length | UDP Checksum |
-----------------------------------

**IFA MD Header:**
-----------------------------------
| Request Vector | Action Vector | Hop Limit | Curr Length |
-----------------------------------

**IFA Metadata Stack:**
-----------------------------------
| IFA Metadata Stack |
-----------------------------------

**IPv6 Header:**
-----------------------------------
| Version | Traffic Class | Flow Label |
-----------------------------------
| Payload Length | Next Hdr = IFA | Hop Limit |
-----------------------------------
| Source IPv6 Address | Destination IPv6 Address |
-----------------------------------

**UDP (Outer VxLAN UDP, IPinIP, GRE Header etc):**
-----------------------------------
| Source Port | Dest Port = VXLAN Port |
-----------------------------------
| UDP Length | UDP Checksum |
-----------------------------------

**IFA MD Header:**
-----------------------------------
| Request Vector | Action Vector | Hop Limit | Curr Length |
-----------------------------------

**IFA Metadata Stack:**
-----------------------------------
| IFA Metadata Stack |
-----------------------------------
IFA IPv4 IPinIP and GRE Encapsulation

- IP Options are not supported for IFA encap
- Inner IP if has IP options will not be parsed for IFA
- Outer IP if has IP options, will not be parsed for IFA
- Load Balancing hash (if using inner IP L4 fields) may not be able to reach inner L4 fields

• Load Balancing hash (if using GRE IP fields) may not be able to reach GRE L4 fields
• GRE options C,R,K,S are treated as part of GRE Header as they are bit defined presence fields (not a header chain)
IFA IPv4 AH (TCP payload) Encapsulation

- IPv4 options are separate then protocol extension headers (unlike IPv6)
- IFA base header is inserted after the IP header
- MD stack is inserted after the IP AH header (works for both transport and tunnel mode)
- IPv4 IFA layering remains same if protocol extension headers are present are not
- AH payload is not touched so IPSec tunnel strip and forward is easy
IFA IPv4 ESP (TCP payload) Encapsulation

- IPv4 options are separate then protocol extension headers (unlike IPv6)
- IFA base header is inserted after the IP header
- MD stack is inserted after the IP AH header (works for both transport and tunnel mode)
- IPv4 IFA layering remains same if protocol extension headers are present are not
- ESP payload is encrypted and not touched so IPSec tunnel strip and forward is easy
IPv6 Extension Header Processing

IPv6 Header:

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Header:</td>
<td>+---------------------------------+</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Version</td>
<td>Traffic Class</td>
<td>Flow Label</td>
<td>+---------------------------------+</td>
</tr>
<tr>
<td>Payload Length</td>
<td>Next Hdr = UL</td>
<td>Hop Limit</td>
<td>+---------------------------------+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Source IPv6 Address</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Destination IPv6 Address</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Upper Layer Hdr</td>
<td>Extension</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>

Order | Header Type | Next Header Code
--- | --- | ---
1 | Basic IPv6 Header | -
2 | Hop-by-Hop Options | 0
3 | Destination Options (with Routing Options) | 60
4 | Routing Header | 43
5 | Fragment Header | 44
6 | Authentication Header | 51
7 | Encapsulation Security Payload Header | 50
8 | Destination Options | 60
9 | Mobility Header | 135
   | No next header | 59
Upper Layer | TCP | 6
Upper Layer | UDP | 17
Upper Layer | ICMPv6 | 58
Upper Layer | IFA | XXX
### IPv6 Extn Header Data Traffic Example

<table>
<thead>
<tr>
<th>Order</th>
<th>Header Type</th>
<th>Next Header Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Basic IPv6 Header</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>Hop-by-Hop Options</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>Destination Options (with Routing Options)</td>
<td>60</td>
</tr>
<tr>
<td>4</td>
<td>Routing Header</td>
<td>43</td>
</tr>
<tr>
<td>5</td>
<td>Fragment Header</td>
<td>44</td>
</tr>
<tr>
<td>6</td>
<td>Authentication Header</td>
<td>51</td>
</tr>
<tr>
<td>7</td>
<td>Encapsulation Security Payload Header</td>
<td>50</td>
</tr>
<tr>
<td>8</td>
<td>Destination Options</td>
<td>60</td>
</tr>
<tr>
<td>9</td>
<td>Mobility Header</td>
<td>135</td>
</tr>
<tr>
<td></td>
<td>No next header</td>
<td>59</td>
</tr>
</tbody>
</table>

**Upper Layer**

- TCP: 6
- UDP: 17
- ICMPv6: 58
- IFA: XXX
IFA IPv6 Extn Header Data Traffic Example

IPv6 Header with Extension Header:
<table>
<thead>
<tr>
<th>Version</th>
<th>Traffic Class</th>
<th>Flow Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>Payload Length</td>
<td>Next Hdr = 43</td>
<td>Hop Limit</td>
</tr>
</tbody>
</table>

~ Source IPv6 Address ~
~ Destination IPv6 Address ~
| NH = 60. | Extension Header |
| NH = 44. | Extension Header |
| NH = IFA | Extension Header |

IFA Header:
| Ver=2.0 | GNS | NextHdr=UDP/TCP | Flags | Max Length |

UDP/TCP Header:
| UDP/TCP Header |

IFA Metadata Stack:
| IFA Metadata Header and Stack |

TCP Payload:
| L4 Payload |

IPv6/EH1/EH2/AH/IFA/L4/MD Stack/L4Payload
IPv6/EH1/EH2/ESP/IFA/MD Stack/{L4/L4Payload}

- Parse Extension header as is (May need punt to CPU)
- Insert IFA base header if UL header is detected
- Insert metadata after the UL header
- Ipv6 IFa layering remains same if extension headers are present are not
Thank You