Internet Engineering Task Force D. Joachimpillai
Internet-Draft Verizon
Intended status: Informational J. Hadi Salim
Expires: July 17, 2014 Mojatatu Networks
January 13, 2014

# ForCES Inter-FE LFB
# draft-joachimpillai-forces-interfelfb-03

Abstract

Forwarding and Control Element Separation (ForCES) defines an architectural framework and associated protocols to standardize information exchange between the control plane and the forwarding plane in a ForCES Network Element (ForCES NE). RFC5812 has defined the ForCES Model which provides a formal way to represent the capabilities, state, and configuration of forwarding elements(FEs) within the context of the ForCES protocol. More specifically, the model describes the logical functions that are present in an FE, what capabilities these functions support, and how these functions are or can be interconnected. The control elements (CEs) can control the FEs using the ForCES model definition.

The ForCES WG charter has been extended to allow the LFB topology to be across FEs. This documents describes a non-intrusive way to extend the LFB topology across FEs.

Status of This Memo

Copyright Notice



Table of Contents

# 1. Terminology and Conventions

## 1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

## 1.2. Definitions

This document follows the terminology defined by the ForCES Model in [RFC5812]. The required definitions are repeated below for clarity.

FE Model - The FE model is designed to model the logical processing functions of an FE. The FE model proposed in this document includes three components; the LFB modeling of individual Logical Functional Block (LFB model), the logical interconnection between LFBs (LFB topology), and the FE-level attributes, including FE capabilities. The FE model provides the basis to define the information elements exchanged between the CE and the FE in the ForCES protocol [RFC5810].

LFB (Logical Functional Block) Class (or type) - A template that represents a fine-grained, logically separable aspect of FE processing. Most LFBs relate to packet processing in the data path. LFB classes are the basic building blocks of the FE model.

LFB Instance - As a packet flows through an FE along a data path, it flows through one or multiple LFB instances, where each LFB is an instance of a specific LFB class. Multiple instances of the same LFB class can be present in an FE's data path. Note that we often refer to LFBs without distinguishing between an LFB class and LFB instance when we believe the implied reference is obvious for the given context.

LFB Model - The LFB model describes the content and structures in an LFB, plus the associated data definition. XML is used to provide a formal definition of the necessary structures for the modeling. Four types of information are defined in the LFB model. The core part of the LFB model is the LFB class definitions; the other three types of information define constructs associated with and used by the class definition. These are reusable data types, supported frame (packet) formats, and metadata.

LFB Metadata - Metadata is used to communicate per-packet state from one LFB to another, but is not sent across the network. The FE model defines how such metadata is identified, produced, and consumed by the LFBs, but not how the per-packet state is

implemented within actual hardware. Metadata is sent between the FE and the CE on redirect packets.

ForCES Component - A ForCES Component is a well-defined, uniquely identifiable and addressable ForCES model building block. A component has a 32-bit ID, name, type, and an optional synopsis description. These are often referred to simply as components.

LFB Component - An LFB component is a ForCES component that defines the Operational parameters of the LFBs that must be visible to the CEs.

LFB Topology - LFB topology is a representation of the logical interconnection and the placement of LFB instances along the data path within one FE. Sometimes this representation is called intra-FE topology, to be distinguished from inter-FE topology. LFB topology is outside of the LFB model, but is part of the FE model.

FE Topology - FE topology is a representation of how multiple FEs within a single network element (NE) are interconnected. Sometimes this is called inter-FE topology, to be distinguished from intra-FE topology (i.e., LFB topology). An individual FE might not have the global knowledge of the full FE topology, but the local view of its connectivity with other FEs is considered to be part of the FE model.

Service Graph - A directed graph of LFB instances whose composition delivers a packet service.

## 2. Introduction

In the ForCES architecture, a packet service can be modelled by composing a graph of one or more LFB instances. The reader is refered to the details in the ForCES Model [RFC5812].

The FEObject LFB capabilities in the ForCES Model [RFC5812] define component ModifiableLFBTopology which, when advertised as true by the FE, implies FE is capable of modifying the LFB graph. In such a case, the table SupportedLFBs contains information about each supported LFB class that the FE supports. For each LFB class supported, additional information of how an LFB class may be connected to other LFBs is advertised. The advertised rules describe which LFB classes a specified LFB class may succeed or precede in an LFB topology. The capability of an FE can be queried by the CE upon association.

The CE may create a packet service by describing LFB instance graph connections via updating the FEOBject LFBTopology component. The created topology contains information about each inter-LFB link within the FE (each link is described in an LFBLinkType dataTypeDef). The LFBLinkType component contains sufficient information to identify precisely the end points of a link of a service graph.

Often there are requirements for the packet service graph to cross FE boundaries. This could be from a desire to scale the service or need to interact with LFBs which reside in a separate FE (eg lookaside interface to a shared TCAM, an interconnected chip, or as coarse grained functionality as an external NAT FE box being part of the service graph etc).

Given that the ForCES inter-LFB architecture calls out for ability to pass metadata between LFBs, it is imperative to define mechanisms to allow passing the metadata between inter-FE LFBs (given that packet data passing is already taken care of).

The new ForCES charter allows the LFB links in a topology to be across multiple FE (inter-FE connectivity).

This document describes extending the LFB topology across FEs i.e inter-FE connectivity without needing any changes to the ForCES definitions. It focusses on using Ethernet as the interconnection as a starting point while leaving room for other protocols (such as directly on top of IP, UDP, VXLAN, etc). Note as of this publication implementation is being evaluated and this documentation will be updated in the future to reflect gained experience.

## 3. Problem Scope And Use Cases

The scope of this document is to solve the challenge of passing ForCES defined metadata and exceptions across FEs (be they physical or virtual). To illustrate the problem scope we present two use cases where we start with a single FE running all the functionality then split it into multiple FEs.

### 3.1. Basic Router

A sample LFB topology Figure 1 demonstrates a service graph for delivering basic IPV4 forwarding service within one FE. Note: although the diagram shows LFB classes connecting in the graph in reality it is a graph of LFB class instances that are inter-connected.

Since the illustration is meant only as an exercise to showcase how data and metadata is sent down or upstream on a graph of LFBs, it

abstracts out any ports in both directions and talks about a generic ingress and egress LFB. Again, for illustration purposes, the diagram does not show expection or error paths. Also left out are details on Reverse Path Filtering, ECMP, multicast handling etc. In other words, this is not meant to be a complete description of an IPV4 forwarding application; for a more complete example, please refer to the LFBlib document [RFC6956] .

The output of the ingress LFB(s) coming into the IPv4 Validator LFB will have both the IPV4 packets and, depending on the implementation, a variety of ingress metadata such as offsets into the different headers, any classification metadata, physical and virtual ports encountered, tunnelling information etc. These metadata are lumped together as "ingress metadata".

Once the IPV4 validator vets the packet (example ensures that no expired TTL etc), it feeds the packet and inherited metadata into the IPV4 unicast LPM LFB.

```
                      +----+
                      |    |
           IPV4 pkt   |    | IPV4 pkt     +-----+             +---+
       +------------->|    |------------->|     |             |   |
       |  + ingress   |    | + ingress    |IPv4 |   IPV4 pkt  |   |
       |   metadata   |    | metadata     |Ucast|------------>|   |--+
       |              +----+              |LPM  |  + ingress  |   |  |
     +-+-+             IPv4               +-----+  + NHinfo   +---+  |
     |   |             Validator                   metadata   IPv4   |
     |   |             LFB                                    NextHop|
     |   |                                                     LFB   |
     |   |                                                           |
     |   |                                                  IPV4 pkt
     |   |                                               + {ingress
     +---+                                                  + NHdetails}
     Ingress                                                metadata |
      LFB                                +-------+                   |
                                         |Egress |                   |
                                      <--|LFB    |<------------------+
                                         +-------+
```

Figure 1: Basic IPV4 packet service LFB topology

The IPV4 unicast LPM LFB does a longest prefix match lookup on the IPV4 FIB using the destination IP address as a search key. The result is typically a next hop selector which is passed downstream as metadata.

The Nexthop LFB receives the IPv4 packet with an associated next hop info metadata. The NextHop LFB consumes the NH info metadata and derives from it a table index to look up the next hop table in order to find the appropriate egress information. The lookup result is used to build the next hop details to be used downstream on the egress. This information may include any source and destination information (MAC address to use, if ethernet;) as well egress ports. [Note: It is also at this LFB where typically the forwarding TTL decrement and IP checksum recalculation occurs.]

The details of the egress LFB are considered out of scope for this discussion. Suffice it is to say that somewhere within or beyond the Egress LFB the IPV4 packet will be sent out a port (ethernet, virtual or physical etc).

### 3.1.1. Distributing The LFB Topology

Figure 2 demonstrates one way the router LFB topology in Figure 1 may be split across two FEs (eg two ASICs). Figure 2 shows the LFB topology split across FEs after the IPV4 unicast LPM LFB.

```
      FE1
    +-------------------------------------------------------------+
    |                            +----+                           |
    | +----------+               |    |                           |
    | | Ingress  |    IPV4 pkt   |    | IPV4 pkt     +-----+      |
    | |  LFB     |+------------->|    |------------->|     |      |
    | |          |  + ingress    |    | + ingress    |IPv4 |      |
    | +----------+    metadata   |    |   metadata   |Ucast|      |
    |      ^                     +----+              |LPM  |      |
    |      |                      IPv4               +-----+      |
    |      |                     Validator              |         |
    |                             LFB                   |         |
    +---------------------------------------------------|---------+
                                                        |
                                                   IPv4 packet +
                                                 {ingress + NHinfo}
                                                     metadata
      FE2                                               |
    +---------------------------------------------------|---------+
    |                                                   V         |
    |             +--------+                       +--------+     |
    |             | Egress |     IPV4 packet       | IPV4   |     |
    |       <-----|  LFB   |<----------------------|NextHop |     |
    |             |        |{ingress + NHdetails}  | LFB    |     |
    |             +--------+      metadata         +--------+     |
    +-------------------------------------------------------------+
```

Figure 2: Split IPV4 packet service LFB topology

Some proprietary inter-connect (example Broadcom Higig over XAUI (XXX: ref needed)) maybe exist to carry both the IPV4 packet and the related metadata between the IPV4 Unicast LFB and IPV4 NextHop LFB across the two FEs.

The purpose of the inter-FE LFB is to define standard mechanisms for interconnecting FEs and for that reason we are not going to touch anymore on proprietary chip-chip interconnects other than state the fact they exist and that it is feasible to have translation to and from proprietary approaches.  The focus is going to stick to FE-FE interconnect where the FE could be physical or virtual and the interconnecting technology runs a standard protocol such as ethernet, IP or other protocols on top of IP.

## 3.2. Arbitray Network Function

In this section we show an example of an arbitrary network function which is more coarse grained in terms of functionality.  Each Network function may constitute more than one LFB.

```
  FE1
+--------------------------------------------------------------+
|                            +----+                            |
| +----------+               |    |                            |
| | Network  |   pkt         |NF2 |    pkt       +-----+       |
| | Function |+------------->|    |------------->|     |       |
| |    1     |  + NF1        |    | + NF1/2      |NF3  |       |
| +----------+    metadata   |    |   metadata   |     |       |
|      ^                     +----+              |     |       |
|      |                                         +-----+       |
|      |                                            |          |
|                                                   |          |
+---------------------------------------------------|----------+
                                                    V
```

Figure 3: A Network Function Service Chain within one FE

The setup in Figure 3 is atypical of most packet processing boxes where we have functions like DPI, NAT, Routing, etc connected in such a topology to deliver a packet processing service to flows.

### 3.2.1. Distributing The Arbitray Network Function

The setup in Figure 3 can be split out across 3 FEs instead as demonstrated in Figure 4. This could be motivated by scale out reasons or because different vendors provide different functionality which is plugged-in to provide such functionality. The end result is to have the same packet service delivered to the different flows passing through.

```
  FE1                        FE2
  +----------+               +----+               FE3
  | Network  |   pkt         |NF2 |    pkt       +-----+
  | Function |+------------->|    |------------->|     |
  |    1     |  + NF1        |    | + NF1/2      |NF3  |
  +----------+    metadata   |    |   metadata   |     |
       ^                     +----+              |     |
       |                                         +-----+
                                                    |
                                                    V
```

Figure 4: A Network Function Service Chain Distributed Across Multiple FEs

## 4. Proposal Overview

We address the inter-FE connectivity by proposing an inter-FE LFB. Using an LFB implies no change to the basic ForCES architecture in the form of the core LFBs (FE Protocol or Object LFBs). This design choice was made after considering an alternative approach that would have required changes to both the FE Object capabilities (SupportedLFBs) as well LFBTopology component to describe the inter-FE connectivity capabilities as well as runtime topology of the LFB instances.

### 4.1. Inserting The Inter-FE LFB

The distributed LFB topology described in Figure 2 is re-illustrated in Figure 5 to show the topology location where the inter-FE LFB would fit in.

```
      FE1
    +-------------------------------------------------------------+
    | +----------+               +----+                           |
    | | Ingress  |    IPV4 pkt   |    | IPV4 pkt     +-----+      |
    | |  LFB     |+------------->|    |------------->|     |      |
    | |          |  + ingress    |    | + ingress    |IPv4 |      |
    | +----------+    metadata   |    |   metadata   |Ucast|      |
    |      ^                     +----+              |LPM  |      |
    |      |                      IPv4               +-----+      |
    |      |                     Validator              |         |
    |      |                      LFB                   |         |
    |      |                                  IPv4 pkt + metadata |
    |      |                        {ingress + NHinfo + InterFEid}|
    |      |                                            |         |
    |                                              +----V----+    |
    |                                              | InterFE |    |
    |                                              |   LFB   |    |
    |                                              +---------+    |
    +---------------------------------------------------|---------+
                                                        |
                                         IPv4 packet and metadata
                                {ingress + NHinfo + Inter FE info}
     FE2                                                |
    +---------------------------------------------------|---------+
    |                                              +----V----+    |
    |                                              | InterFE |    |
    |                                              |   LFB   |    |
    |                                              +---------+    |
    |                                                   |         |
    |                                         IPv4 pkt + metadata |
    |                                          {ingress + NHinfo} |
    |                                                   |         |
    |             +--------+                       +----V---+     |
    |             | Egress |     IPV4 packet       | IPV4   |     |
    |       <-----|  LFB   |<----------------------|NextHop |     |
    |             |        |{ingress + NHdetails}  | LFB    |     |
    |             +--------+      metadata         +--------+     |
    +-------------------------------------------------------------+
```

Figure 5: Split IPV4 forwarding service with Inter-FE LFB

As can be observed in Figure 5, the same details passed between IPV4 unicast LPM LFB and the IPV4 NH LFB are passed to the egress side of the Inter-FE LFB. In addition an index for the inter-FE LFB (interFEid) is passed as metadata.

The egress of the inter-FE LFB uses the received Inter-FE index (InterFEid metadata) to select details for encapsulation towards the

neighboring FE. These details will include what the source and destination FEID to be communicated to the neighboring FE. In addition the original metadata, any exception IDs may be passed along with the original IPV4 packet.

On the ingress side of the inter-FE LFB the received packet and its associated details are used to decide the graph continuation i.e which FE instance is to be passed the packet and what of the original metadata and exception IDs. In the illustrated case above, an IPV4 Nexthop LFB instance metadata is passed.

The ingress side of the inter-FE LFB consumes some of the information passed (eg the destination FEID) and passes on the IPV4 packet alongside with the ingress + NHinfo metadata to the IPV4 NextHop LFB as was done earlier in both Figure 1 and Figure 2.

## 4.2. Inter-FE connectivity

We describe the suggested encapsulation format (Figure 6) extended from the ForCES redirect packet format. We expect that for any transport mechanism used, that a description of how the different fields will be encapsulated to be explained. We provide a description of how ethernet encapsulation will be used in this case in Section 4.2.1.

```
        +-- Main ForCES header
        |   |
        |   +---- msg type = REDIRECT
        |   +---- Destination FEID
        |   +---- Source FEID
        |   +---- NEID (first word of Correlator)
        |
        +-- T = ExceptionID-TLV
           |   |
           |   +-- +-Exception Data ILV (I = exceptionID , L= length)
           |   |   |  |
           |   |   |  +----- V= Metadata value
           |   .   |
           |   .   |
           |   .   +-Exception Data ILV
           .
           |
           +-- T = METADATA-TLV
           |   |
           |   +-- +-Meta Data ILV (I = metaid, L= length)
           |   |   |  |
           |   |   |  +----- V= Metadata value
           |   .   |
           |   .   |
           |   .   +-Meta Data ILV
           .
           +-- T = REDIRECTDATA-TLV
               |
               +--  Redirected packet Data
```

Figure 6: Packet format suggestion

o  The ForCES main header as described in RFC5810 is used as a fixed header to describe the Inter-FE encapsulation.

   *  The Source ID field is mapped to the originating FE and the destination ID is mapped to the destination FEID.

   *  The first 32 bits of the correlator field are used to carry the NEID.  The 32-bit NEID defaults to 0.

o  The ExceptionID TLV carries one or more exception IDs within ILVs. The I in the ILV carries a globally defined exceptionID as per-ForCES specification defined by IANA.  This TLV is new to ForCES and sits in the global ForCES TLV namespace.

o  The METADATA and REDIRECTDATA TLV encapsulations are taken directly from [RFC5810] section 7.9.

### 4.2.1. Inter-FE Ethernet connectivity

It is expected that a variety of transport encapsulations would be applicable to carry the format described in Figure 6. In such a case, a description of a mapping to intepret the inter-FE details and translate into proprietary or legacy formatting would need to be defined. For any mapping towards these definitions a different document to describe the mapping, one per transport, is expected to be defined.

In this specific document, we describe a format that is to be used over Ethernet. An ethernet type (To be defined) will be used to imply that a wire format is carrying an inter-FE LFB packet.

XXX: The finer details on what the source and destination MAC address selection are left out for the next draft release. Also left out are any load balancing/multi-pathing activities across selections of destinations FEs.

```
*--+ Ethernet header (ethertype = XXXX)
   |
   +-- Main ForCES header
   |   |
   |   +---- msg type = REDIRECT
   |   +---- Destination FEID
   |   +---- Source FEID
   |   +---- NEID -- Correlator first word
   |   +---- {frag count, frag total}
   |
   |
   +-- T = ExceptionID-TLV
   |   |
   |   +-- +-Exception Data ILV (I = exceptionID , L= length)
   |   |   |  |
   |   |   |  +----- V= Metadata value
   |   .   |
   |   .   |
   |   .   +-Exception Data ILV
   .
   |
   +-- T = METADATA-TLV
   |   |
   |   +-- +-Meta Data ILV (I = metaid, L= length)
   |   |   |  |
   |   |   |  +----- V= Metadata value
   |   .   |
   |   .   |
   |   .   +-Meta Data ILV
   .
   +-- T = REDIRECTDATA-TLV
       |
       +--  Redirected packet Data
```

Figure 7: Packet format suggestion

Notice the next 32 bits of the correlator are used for accounting of fragmentation.

#### 4.2.1.1. Inter-FE Ethernet Connectivity Issues

There are several issues that may arise due to using direct ethernet encapsulation.

o The frame may end up being larger than the MTU. This is to be expected in particular where one LFB instance requires assembling for example a full IPV4 message before passing it downstream to

another FE's LFB instance for further processing. There are several possible solutions:

* One possible solution is to use large MTUs; however, even that will have limits since the the ethernet frames could grow arbitrarily large with increasing metadata being encapsulated.

* An alternative approach is to add a fragmentation detail in the encapsulation. A simple approach is to have the inter-FE LFB (egress) add another header which submits total count of fragments and the fragment number of the submitted packet. The ingress of the inter-FE LFB will keep track of the fragments, assemble them as well as have a timer to discard outstanding fragments.

* A third option is to limit the amount of metadata that could be transmitted so that the frame is sub-MTU size in presence of large MTU values. It will mean to add knobs to filter out or select which metadata gets encapsulated.

* A fourth option is to use a transport that provides fragmentation services (such as IP).

o The frame may be dropped if there is congestion on the receiving FE side. This may necessitate a retransmission mechanism to be built in. One approach to mitigate this issue is to make sure that inter-FE LFB frames receive the highest priority treatment when scheduled on the wire. A more common approach used in tunneling is to not care and let the packet originator to resend if they care about reliability.

We opt for the option of using the first suggestion where the sending side when fragmenting packets accounts for them as a count of total. The second 32 bit part of the ForCES correlator is split into two 16-bit fields for this activity: The first 16bit is for the fragment number and the second one is for the fragment total. As an example if there were two fragments, the first one would be: 1 of 2 and the last one 2 of 2. XXX: Outstanding question still is if we only fragment the data and not the other fields? It seems the first frame will always have all the metadata + exception TLVs and subsequent TLVs will have metadata.

## 5. Detailed Description of the inter-FE LFB

The inter-FE LFB has two LFB input ports and three LFB output ports.

```
                   +-----------------+
                   |                 |
    Encapsulated   |             OUT2+--> decapsulated Packet + metadata
    -------------->|IN2              |      + exception IDs
    Packet         |                 |
                   |                 |
    raw Packet +   |             OUT1+--> encapsulated Packet
    -------------->|IN1              |
    exceptionIDs+  |                 |
    Metadata +     |    EXCEPTIONOUT +--> Errorid, packet + metadata
                   |                 |
                   +-----------------+
```

Figure 8: Inter-FE LFB

## 5.1. Data Handling

The Inter-FE LFB may be positioned at the egress of an FE. In such a case it receives via port IN1, raw packet, metadata, and exception IDs. The InterFEid metadatum MAY be present on the incoming raw data. The processed encapsulated packet will go out on either LFB port OUT1 to a downstream LFB or EXCEPTIONOUT port in the case of a failure.

The Inter-FE LFB may be positioned at the ingress of an FE. In such a case it receives, via port IN2, an encapsulated packet. Successful processing of the packet will result in a raw packet with associated metadata and exception IDs going downstream to an LFB connected on OUT2. On failure the data is sent out EXCEPTIONOUT.

An implementation may have one one or more Ingress or egress inter-FE LFB instances. As an example, there could be one instance for the ingress side and a second instance for the egress side. An alternative approach maybe to have an ingress and egress instance per port.

The Inter-FE LFB uses the InterFEid metadatum when on an egress of an FE to lookup the NextFE table. The interFEid will be generated by an upstream LFB instance (i.e one preceeding the Inter-FE LFB). The output result constitutes a matched table row which has the InterFEinfo details i.e. the tuple {NEID,Destination FEID,Source FEID, metafilters, exceptionfilters}. The two filter lists define which Metadatum and/or exceptionids are to be passed to the neighboring FE. It is expected that zero configuration is needed; in the absence of the InterFEid metadatum, default behavior will be utilized.

### 5.1.1. Egress Processing

The InterFEid is used to lookup NextFE table. If lookup is successful, the inter-FE LFB will:

- o add the NEID data from the lookup result
- o walk the passed metadatum, apply the filters and encapsulate allowed ones them within METADATA-TLV as separate ILVs. The InterFEid is never passed.
- o walk all the passed exceptionIDs, apply the filters and encapsulate all allowed exception IDs within EXCEPTION-TLV header (as ILVs).
- o Encapsulate the data, if present, in REDIRECTDATA-TLV
- o XXX: We need to describe the fragmentation handling in next update.

The resulting packet is sent to the LFB instance connected to the OUT1 LFB port.

In the case of a failed lookup or a zero-value InterFEid, or absence of InterFEid, the default inter-FE LFB processing will:

- o Set the NEID to 0.
- o walk all the passed metadatum and encapsulate into the METADATA-TLV all metadatum. The InterFEid is never passed.
- o walk all the passed exceptionIDs and encapsulate each exceptionID within the EXCEPTION-TLV.
- o Encapsulate the data, if present, in REDIRECTDATA-TLV

The resulting packet is sent to the LFB instance connected to the OUT1 LFB port.

### 5.1.2. Ingress Processing

An inter-FE packet is recognized by looking at the etherype.

In the ingress processing, the approriate inter-FE LFB instance receives an encapsulated packet and extracts the packet data, metadata, and exception IDs. This data is then passed downstream to the next programmed LFB instance.

In the case of processing failure of either ingress or egress positioning of the LFB, the packet and metadata are sent out the EXCEPTIONOUT LFB port with proper error id (XXX: More description to be added).

XXX: We need to describe the fragmentation handling after a WG discussion.

## 5.2. Metadata

A single (to be define from IANA space) metadatum, InterFEid, is defined.

## 5.3. Components

There is a single optional LFB component populated by the CE. The component is an array known as the NextFE table. Each row of the table constitutes the columns with {NEID,Destination FEID,Source FEID,array of allowed Metaids, array of allowed exception ids}. The table is looked up by a 32 bit index passed from an upstream LFB class instance in the form of InterFEid metadatum.

The CE programs LFB instances in a service graph that require inter-FE connectivity with InterFEid values to correspond to the inter-FE LFB NextFE table entries to use.

## 5.4. Capabilities

XXX: If we support multiple encapsulation methods(other than ethernet), then we could use capabilities to advertise them as different possibilities. It is envisioned then that the NextFE table row will have column indicating to the inter-FE LFB how to encapsulate the different matches. Alternatively this could be left up to the LFB connected in the output port.

## 5.5. Events

TBA

## 5.6. Inter-FE LFB XML

TBA

## 6. Acknowledgements

The authors would like to thank Joel Halpern and Dave Hood for the stimulating discussions.

## 7. IANA Considerations

This memo includes one requests to IANA for InterFE Metaid.

## 8. Security Considerations

TBD

## 9. References

### 9.1. Normative References

[RFC5810] Doria, A., Hadi Salim, J., Haas, R., Khosravi, H., Wang, W., Dong, L., Gopal, R., and J. Halpern, "Forwarding and Control Element Separation (ForCES) Protocol Specification", RFC 5810, March 2010.

[RFC5812] Halpern, J. and J. Hadi Salim, "Forwarding and Control Element Separation (ForCES) Forwarding Element Model", RFC 5812, March 2010.

### 9.2. Informative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

Authors' Addresses

Damascane M. Joachimpillai
Verizon
60 Sylvan Rd
Waltham, Mass. 02451
USA

Email: damascene.joachimpillai@verizon.com

Jamal Hadi Salim
Mojatatu Networks
Suite 400, 303 Moodie Dr.
Ottawa, Ontario K2H 9R4
Canada

Email: hadi@mojatatu.com