INTERNET-DRAFT
Intended Status: Standards Track
Expires: April 30, 2015

Luyuan Fang
Vijay Gill
Microsoft
Fabio Chiussi

October 27, 2014

MPLS-Based Hierarchical SDN for Hyper-Scale DC/Cloud
draft-fang-mpls-hsdn-for-hsdc-00

Abstract

This document describes Hierarchical SDN (HSDN), an architectural solution to scale the Data Center (DC) and Data Center Interconnect (DCI) networks to support tens of millions of physical underlay endpoints, while efficiently handling both ECMP and any-to-any end-to-end Traffic Engineered (TE) traffic. HSDN achieves massive scale using surprisingly small forwarding tables in the network nodes and brings key simplifications in the control plane as well. The HSDN forwarding architecture is based on four main concepts: 1. Dividing the DC and DCI in a hierarchically-partitioned structure; 2. Assigning groups of Underlay Border Nodes in charge of forwarding within each partition; 3. Constructing HSDN MPLS label stacks to identify the endpoints according to the HSDN structure; and 4. Forwarding using the HSDN MPLS labels.

http://www.ietf.org/shadow.html

Table of Contents

1. Introduction

With the growth in the demand for cloud services, the end-to-end cloud network, which includes Data Center (DC) and Data Center Interconnect (DCI) networks, has to scale to support millions to tens of millions of underlay network endpoints. These endpoints can be bare metal servers, virtualized servers, or physical and virtualized network functions and appliances.

The scalability challenge is twofold: 1. Being able to scale using low-cost network nodes while achieving high resource utilization in the network; and 2. Being able to scale at low operational and computational complexity while supporting Equal-Cost Multi-Path (ECMP) and any-to-any Traffic Engineering (TE).

An important set of scalability issues to resolve comes from the potential explosion of the routing tables in the network nodes as the number of underlay network endpoints increases. Current commodity switches have relatively small routing and forwarding tables. For example, the typical Forwarding Information Base (FIBs) and Label Forwarding Information Base (LFIBs) tables in current low-cost network nodes contain 16K or 32K entries. These small sizes are clearly insufficient to support entries for all the endpoints in the hyper-scale cloud. Address aggregation is used to ameliorate the problem, but the scalability challenges remain, since the dynamic and elastic environment in the DC/cloud often brings the need to handle finely granular prefixes in the network.

Other factors contribute to the FIB/LFIB explosion. For example, in a typical DC using a fat Clos topology, even the support of ECMP load balancing may become an issue if the individual outgoing paths belonging to an ECMP group carry different outgoing labels.

Another key scalability issue to resolve is the complexity of certain desired functions that should be supported in the network, the most prominent one being TE. Currently, any-to-any server-to-server TE in the DC/DCI is simply unfeasible, as path computation and bandwidth allocation at scale, an NP-complete problem, becomes rapidly unmanageable. Furthermore, the forwarding state needed in the network nodes for TE tunnels contributes in a major way to the explosion of the LFIBs.

Other major scalability issues are related to the efficient creation, management, and use of tunnels, for example the configuration of protection paths for fast restoration.

Many additional scalability issues in terms of operational and computational complexity need to be resolved in order to scale the control plane and the network state. In particular, the controller-centric approach of Software Defined Networks (SDNs), which is increasingly being accepted as "the way to build the next generation clouds," in order to be scalable, requires appropriate scalability solutions in order to take full advantage of the potential benefits of SDN.

Finally, the underlay network architecture should offer certain capabilities to facilitate the support of the demand of the overlay network.

In this document, we present Hierarchical SDN (HSDN), a set of solutions for all these scalability challenges in the underlay network, both in the forwarding and in the control plane. Although HSDN can be used in principle with any forwarding technology, it has been designed to leverage Multi Protocol Label Switching (MPLS)-based forwarding [RFC3031], using label stacks [RFC3032] constructed according to the HSDN structure.

HSDN achieves massive scale using surprisingly small LFIBs in the network nodes, while supporting both ECMP and any-to-any end-to-end TE traffic. HSDN also brings important simplifications in the control plane and in the architecture of the SDN controller.

The HSDN forwarding architecture is based on four main concepts: 1. Dividing the DC and DCI in a hierarchically-partitioned structure; 2. Assigning groups of Underlay Border Nodes in charge of forwarding within each partition; 3. Constructing HSDN MPLS label stacks to identify the end points according to the HSDN structure; and 4. Forwarding using the HSDN MPLS labels.

HSDN is designed to allow the physical decoupling of the control and forwarding, and have the LFIBs configured by a controller according to the SDN approach. However, it is also meant to support the traditional distributed routing and label distribution protocol approach, which may be particularly useful during technology migration.

1.1. Terminology

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 RFC 2119 [RFC2119].

Term	Definition
BGP	Border Gateway Protocol
DC	Data Center
DCGW	DC Gateway (Border Leaf)
DCI	Data Center Interconnect
DID	Destination Identifier
ECMP	Equal Cost MultiPathing
FIB	Forwarding Information Base
HSDN	Hierarchical SDN
LDP	Label Distribution Protocol
LFIB	Label Forwarding Information Base
LN	Leaf Node
MPLS	Multi-Protocol Label Switching
PID	Path Identifier
SDN	Software Defined Network
SN	Spine Node
SVR	Server
UP	Underlay Partition
UPBG	Underlay Partition Border Group
UPBG	Underlay Partition Border Node
TE	Traffic Engineering
ToR	Top-of-Rack switch
TR	Top-of-Rack switch
VN	Virtual Network
VM	Virtual Machine
WAN	Wide Area Network

In this document, we also use the following terms.

- End device: A physical device attached to the DC/DCI network. Examples of end devices include bare metal servers, virtualized servers, network appliances, etc.

- Level: A layer in the hierarchy of underlay partitions in the HSDB architecture.

- Overlay Network (ON): A virtualized network that provides Layer 2 or Layer 3 virtual network services to multiple tenants. It is implemented over the underlay network.

- Path Label (PL): A label used for MPLS-based HSDN forwarding in the underlay network.

- Row: A row of racks where end devices reside in a DC.

- Tier: One of the layers of network nodes in a multi-layer Clos-based topology.

o Underlay Network (UN): The physical network that provides the connectivity among physical end devices. It provides transport for the overlay network traffic.

o Underlay Partition (UP): A logical portion of the underlay network designed according to the HSDN architecture. Underlay partitions are arranged in a hierarchy consisting of multiple levels.

o VN Label (VL): A label carrying overlay network traffic. It is encapsulated in the underlay network in a stack of path labels constructed according to the HSDN forwarding scheme.

1.2. DC and DCI Reference Model

Here we show the typical structure of the DC and DCI, which we use in the rest of this document to describe the HSDN architecture. We also introduce a few commonly used terms to assist in the explanation.

Figure 1 illustrates multiple DCs interconnected by the DCI/WAN.

```
                     +-------------+
                     |             |
                     |     DC      |
                     |             |
                     +-------------+
                               \-----.
                               (       ')
                           .--(.,      '.---.
                          (     '      '     )
                         (       DCI/WAN      )
                          (.                .)
                           (     (       .)  \
                            /'--' '-''---'  +-------------+
                 +-------------+            |             |
                 |             |            |     DC      |
                 |     DC      |            |             |
                 |             |            +-------------+
                 +-------------+
```

Figure 1. DCIWAN interconnecting multiple DCs.

Figure 2 below illustrates the typical structure of a Clos-based DC fabric.

```
                                 +----+   +----+
  DCGW                 +---------+ GW +   + GW +---------+
                       |         ++--++\ /++--++         |
                       | +--------|--|--\--\  |          |
                       | |        |  \-/-\-|--|--------+ |
                     +-+-++     +-+--+/   \+--+-+     ++-+-+
  Spine    +---------+ SN +     | SN |     | SN +     + SN +---------+
           |         +----+\    ++--++     ++--++\   /+----+         |
           | +---------------\---+  |       |  +---/---------------+ |
           | |        ...      \    |       |    /      ...        | |
          ++-+-+ +----+  +----+ ++--++     ++--++ +----+  +----+ +-+-++
  Leaf    | LN | | LN |  | LN | | LN |     ||LN | | LN |  | LN | | LN |
          ++--++ ++--++  +---++ ++---+     +---++ ++--++  +---++ ++--++
           |   \ /   |    |   \ /   |       |   \ /   |    |   \ /   |
           |   / \   |    |   / \   |       |   / \   |    |   / \   |
           | / ... \ |    | / ... \ |       | / ... \ |    | / ... \ |
         ++++ +--+ ++++ ++++ +--+ ++++    ++++ +--+ ++++ ++++ +--+ ++++
  ToR    |TR| |TR| |TR| |TR| |TR| |TR|    |TR| |TR| |TR| |TR| |TR| |TR|
         +--+ +--+ +--+ +--+ +--+ +--+    +--+ +--+ +--+ +--+ +--+ +--+
          |    |    |    |    |    |       |    |    |    |    |    |
         +--+ +--+ +--+ +--+ +--+ +--+    +--+ +--+ +--+ +--+ +--+ +--+
  Server +--+ +--+ +--+ +--+ +--+ +--+    +--+ +--+ +--+ +--+ +--+ +--+
  rack   +--+ +--+ +--+ +--+ +--+ +--+    +--+ +--+ +--+ +--+ +--+ +--+
         +--+ +--+ +--+ +--+ +--+ +--+    +--+ +--+ +--+ +--+ +--+ +--+
```

Figure 2. Typical Clos-based DC fabric topology.

Note: Not all links are shown in Figure 2.

The DC fabric shown in Figure 2 uses what is known as a spine and leaf architecture with a multi-stage Clos-based topology interconnecting multiple tiers of network nodes. The DC Gateways (DCGWs) connect the DC to the DCI/WAN. The DCGW connect to the Spine Nodes (SNs), which in turn connect to the Leaf Nodes (LFs). The Leaf Nodes connect to the Top-of-Rack switches (ToRs). Each ToR typically resides in a rack (hence the name) accommodating a number of servers connected to their respective ToR. The servers may be bare metal or virtualized.

Each tier of switches and the connectivity between switches is designed to offer a desired capacity and provide sufficient bandwidth to the servers and end devices.

The precise topology and connectivity between the tiers of switches depends on the specific design of the DC. More or less tiers of switches (spines or leaves) or asymmetric topologies, not shown in the figure, may be used. A precise description of the topology and its design criteria is out of the scope of this document.

What’s relevant for this document is the fact that a typical large-scale DC topology does not have all the tiers fully connected to the adjacent tiers. In other words, not all network nodes in a tier are connected to all the network nodes in the adjacent tiers. This is especially true for the tiers closer to the endpoints, and is due to the sheer number of connections and devices and the physical constraints of the DC, which makes it impractical, uneconomical, and ultimately unnecessary to use a fully connected Clos-based topology.

The connectivity is typically organized following an aggregation/multiplexing connectivity architecture that consolidates traffic from the edges into the leafs and spines.

2. Requirements

2.1. MPLS-Based HSDN Design Requirements

The following are the key design requirements for HSDN solutions.

1) MUST support millions to tens of millions of underlay network endpoints in the DC/DCI.

2) MUST use very small LFIB sizes (e.g., 16K or 32K LFIB entries) in all network nodes.

3) MUST support both ECMP and any-to-any, end-to-end, server-to-server TE traffic.

4) MUST support ECMP traffic load balancing using a single forwarding entry in the LFIBs per ECMP group.

5) MUST require IP lookup only at the network edges.

6) MUST support encapsulation of overlay network traffic, and support any network virtualization overlay technology.

7) MUST support control plane using both SDN controller approach, and the traditional distributed control plane approach using any label distribution protocols.

2.2. Hardware Requirements

The following are the hardware requirements to support HSDN.

1) The server NICs MUST be able to push a HSDN label stack consisting of as many path labels as levels in the HSDN hierarchical partition (e.g., 3 path labels).

2) The network nodes MUST support MPLS forwarding.

3) The network nodes MUST be able perform ECMP on packets carrying a label stack consisting of as many path labels as levels in the HSDN hierarchical partition, plus one or more VN label/header for the overlay network (e.g., 3 path labels + 1 VN label/header).

3. HSDN Architecture - Forwarding Plane

As mentioned above, a primary design requirement for HSDN is to enable scalability of the forwarding plane to tens of millions of network endpoints using very small LFIB sizes in all network nodes in the DC/DCI, while supporting both ECMP and any-to-any server-to-server TE traffic.

The driving principle of the HSDN forwarding plane is "divide and conquer" by partitioning the forwarding task into local and independent forwarding. When designed properly, such an approach enables extreme horizontal scaling of the DC/DCI.

HSDN is based on four concepts:

1) Dividing the underlay network in a hierarchy of partitions;
2) Assigning groups of Underlay Partition Border Nodes (UPBN) to each partition, in charge of forwarding within the corresponding partition;
3) Constructing HSDN label stacks for the endpoint Forward Equivalency Classes (FECs) in accordance with the underlay network partition hierarchy;
4) Configuring the LFIBs in all network nodes and forwarding using the label stacks.

In this section, we explain in detail each of these concepts. Scalability analysis for both ECMP and TE is presented in Section 4. In Section 5, we describe a possible label stack assignment scheme for HSDN.

3.1. Hierarchical Underlay Partitioning

HSDN is based on dividing the DC/DCI underlay network into logical partitions arranged in a multi-level hierarchy.

The HSDN hierarchical partitioning is illustrated in Figure 3.

```
           +----------------------------------------------------------+
  UP       |                          UP0                             |
  Level 0  |                                                          |
           |+---------------------------------+   +------------------+|
           ||                                 |...|                  ||
           +|---------------------------------|---|------------------|+
  UP        |             UP1-0               |   |       UP1-N      |
  Level 1   |                                 |   |                  |
            |+-------------+   +-------------+|   |   +-------------+|
            ||             |...|             ||   |...|             ||
            +|-------------|---|-------------|+   +---|-------------|+
  UP         |   UP2-0-0   |   |   UP2-0-N   |        |  UP2-N-N    |
  Level 2    |             |   |             |        |             |
             +-----+ +-----+   +-----+ +-----+        +-----+ +-----+
             | SVR | | SVR |   | SVR | | SVR |        | SVR | | SVR |
             |-----|-|-----|   |-----|-|-----|        |-----|-|-----|
  Overlay    |VM|VM| |VM|VM|   |VM|VM| |     |        |VM|VM| |     |
  Level      |-----| |-----|   |-----| |     |        |-----| |     |
             |VM|VM| |VM|VM|   |VM|VM| |     |        |VM|VM| |     |
             +-----+ +-----+   +-----+ +-----+        +-----+ +-----+
```

Figure 3. HSDN underlay network hierarchical partitioning of DC/DCI.

The hierarchy consists of multiple levels of Underlay Partitions (UPs). For simplicity, we describe HSDN using three levels of partitioning, but more or less levels can be used, depending on the size and architecture of the overall network, using similar design principles (as shown below, three levels of partitions are sufficient to achieve scalability to tens of millions servers using very small LFIBs).

The levels of partitions are nested into a hierarchical structure. At each level, the combination of all partitions covers the entire DC/DCI topology. In general, within each level, the UPs do not overlap, although there may be design scenarios in which overlapping UPs within a level may be used. The top level (Level 0) consists of a single underlay partition UP0 (the HSDN concept can be extended to multi-partitioned Level 0).

We use the following naming convention for the UPs:

- Partitions at Level i are referred to as UPi (e.g., UP0 for Level 0, UP1 for Level 1, UP2 for Level2, and so on).

- Within each level, partitions are identified by a rightmost sequential number (starting from 1) referring to the corresponding level and a set of sequential number(s) for each partition in a higher level that the specific partition is nested into.

For example, at Level 1, there are N partitions, referred to as UP1-1 to UP1-N.

Similarly, at Level 2, there are M partitions for each Level 1 partitions, for a total of NxM partitions. For example, the Level 2 partitions nested into Level 1 partition UP1-1 are UP2-1-1 to UP2-1-M, while the ones nested into UP1-N are UP2-N-1 to UP2-N-M.

- Note that for simplicity in illustrating the partitioning, we assume a symmetrical arrangement of the partitions, where the number of partitions nested into each partition at a higher level is the same (e.g., all UP1 partitions have M UP2 partitions). In practice, this is rarely the case, and the naming convention can be adapted accordingly for different numbers of partitions.

The following considerations complete the description of Figure 3.

o The servers (bare metal or virtualized) are attached to the bottom UP level (in our case, Level 2). A similar naming convention as the one used for the partitions may be used.

o In Figure 3, we also show an additional Overlay Level. This corresponds to the virtualized overlay network (if any) providing Virtual Networks (VN) connecting Virtual Machines (VMs) and other overlay network endpoints. Overlay network traffic is encapsulated by the HSDN underlay network. The operation of the Overlay Level is out of scope of this document.

The UPs are designed to contain one or more tiers of switches in the DC topology or nodes in the DCI. The key design criteria in defining the partitions at each layer is that they need to follow the "natural" connectivity implemented in the DC/DCI topology. An example is given below to further clarify how the partitions are designed.

3.2. Underlay Partition Border Nodes

Once the HSDN hierarchical partitioning is defined, Underlay Partition Border Nodes (UPBNs) are assigned to each UP. This is illustrated in Figure 4.

```
  +++++++++++++++++++++++++++++++++++++++++++++++++
  |                     UP0              <--------|-----+
  |                                               |     |
  | +++++++++++++++++++++++++++++++++++++++++++++ |     |
  | |   +-----------------------------------+   | |     |
  | |   | +---------+  UPBG1-i  +---------+ |   | |     |
  | |   | | UPBN1-i |    ...    | UPBN1-i | |   | |     |
  | |   | +---------+           +---------+ |   | |     |
  | |   +-----------------------------------+   | |     |
  ++|+++++++++++++++++++++++++++++++++++++++++++|++     |
    |                   UP1              <------|----+  |   +---------+
    | +++++++++++++++++++++++++++++++++++++++++ |    |  +---|   LP0    |
    | | +-----------------------------------+ | |    |      +---------+
    | | | +---------+ UPBG2-i-j +---------+ | | |    +------|   LP1    |
    | | | |UPBN2-i-j|    ...    |UPBN2-i-j| | | |           +---------+
    | | | +---------+           +---------+ | | |    +------|   LP2    |
    | | +-----------------------------------+ | |    |      +---------+
    ++|+++++++++++++++++++++++++++++++++++++++|++    |  +---|   VL     |
      |                 UP2              <----|------+  |   +---------+
      |          +---------------+            |         |
      |          |   SVR-i-j-k   |            |         |
      |          | +-----------+ |            |         |
      |          | |    NVE    |<-------------|---------+
      |          | +-----+-----+ |            |
      +++++++++++| | VM  | VM  | |+++++++++++++
                 | +-----+-----+ |
                 | | VM  | VM  | |
                 +-+-----+-----+-+
```

Figure 4. UBPNs, UBPGs, and label stack assignment.

The UPBNs belong to two partitions in adjacent levels in the hierarchy and they constitute the entry points for traffic from the higher level partition destined to the corresponding lower level partition. As such, they constitute the forwarding end destinations within each partition.

In order to provide sufficient capacity and support traffic load balancing between the levels in the hierarchy, multiple UPBNs are assigned to each partition. The UPBNs for each partition are grouped into an Underlay Partition Border Group (UPBG). As shown below, using an appropriate Label Stack Assignment scheme all UPBNs in a UPBG can be made identical for ECMP traffic forwarding (i.e., the ECMP entries in the LFIBs in all UPBNs in a UPBG are identical). Thus, for ECMP traffic load balancing, all UPBNs belong to the same FEC as far as the higher level partition is concerned. For TE traffic, a desired UPBN within a UPBG group may need to be specified, and thus the UPBNs in a UPBG are not forwarding-wise equivalent.

In practice, the UPs are designed by finding the most advantageous way to partition the DC Clos-based topology and the DCI topology. Within the DC, the UPBNs in each level are subsets of the network nodes in one of the tiers that form the multi-stage Clos architecture. In general, the UPs may internally contain tiers of network nodes that are not UPBNs. A specific design example to further illustrate the HSDN partitioning is provided below.

As explained in more detail below, for forwarding purposes, by partitioning the DC/DCI in this manner and using HSDN forwarding, the UPBNs need to have entries in their LFIBs only to reach destinations in the two partitions to which they belong to (their own corresponding partition and the higher-layer partition to which they nested to). The network nodes inside the UPs only need to have entries in their LFIB to reach the destinations in their partition.

From these considerations, a first design heuristic for choosing the partitioning structure is to keep the number of partitions nested at each level into the higher level relatively small for all levels. For the lowest level, the number of endpoints (servers) in each partition should also be kept to manageable levels. Clearly, the design tradeoff is between the size and the number of partitions at each level. Fortunately, for most practical deployments, it is relatively simple to find a good tradeoff that achieves the desired scalability.

3.2.1. UPBN and UPBG Naming Convention

We use a similar naming convention for the UPBNs and UPBGs as the one used for the UPs:

- UPBNi is a UPBN between partitions at Level(i) and Level(i-1). Similarly for UPBG.

- Within each level, the UPBNs are identified by a set of sequential number(s) equal to the corresponding sequential number(s) of the corresponding partition within that level.

  For example, at Level 1, UPBN1-1 corresponds to partition UP1-1, and connects UP0 with UP1-1. UPBN1-N corresponds to partition UP1-N and connects UP0 with UP1-N, and so on. Similarly for UPBG.

  At Level 2, UPBN2-1-1 corresponds to partition UP2-1-1 and connects UP1-1 with UP2-1-1, and so on. Similarly for UPBG.

  Note that the UPBNs within an UPBGs can be further distinguished using an appropriate naming convention, which is not shown here.

3.2.2. HSDN Label Stack

In MPLS-Based HSDN, an MPLS label stack is defined and used for forwarding. The key notion in HSDN is that the label stack is defined and the labels are assigned in accordance with the hierarchical partitioned structure defined above.

The label stack, shown in Figure 4 above, is constructed as follows.

- The label stack contains as many Path Labels (PLs) as levels in the partition hierarchy.

- Each PL in the label stack is associated to a corresponding level in the partition hierarchy and is used for forwarding at that level.

  In the scenario of Figure 4, PL0 is associated to Level 0 and is used to forward to destination in UP0, PL1 is associated to Level 1 and is used to forward to destinations in any UP1 partitions, and PL2 is associated to Level 2 and is used to forward to destinations in any UP2 partitions.

- A VN Label (VL) is also shown in the label stack in Figure 4. This label is associated to the Overlay Level and is used to forward in the overlay network. The VL is simply encapsulated in the label stack and transported in the HSDN underlay network. The details of the VL processing within the overlay network are out of scope of this document.

Each endpoint in the DC/DCI is identified by a corresponding label stack. For a given endpoint, the label stack is constructed in such a way that the PLO specifies the UP1 to which the endpoint is attached to, the PL1 specifies the UP1 to which the endpoint is attached to, and the PL2 specifies the FEC in the UP2 corresponding to the endpoint.

A scheme to assign the PL labels in the HSDN label stack is described below.

3.2.3. HSDN Design Example

We use an example to further explain the HSDN design criteria to define the hierarchically-partitioned structure of the DC/DCI. We use the same design example in the Scalability Analysis section below to show the LFIB sizing with ECMP and TE traffic.

To summarize some of the design heuristics for the HSDN underlay partitions:

- The UPs should be designed to follow the "natural" connectivity topology in the DC/DCI.

- The number of partitions at each level nested into the higher level should be relatively small (since they are FEC entries in the LFIBs in the network nodes in the corresponding levels).

- The number of endpoints (servers) in each partition in the lowest level should be relatively small (since they are FEC entries in the LFIBs in the network nodes in the lowest level).

- The number of levels should be kept small (since it corresponds to the number of path labels in the stack).

- The number of tiers in each partition in each level should be kept small. This is due to the multiplicative fanout effect for TE traffic (explained below), which has a major impact on the LFIB size needed to support any-to-any server-to-server TE.

The HSDN forwarding plane design consists in finding the best tradeoff among these contrasting objectives. Although the optimal design choices ultimately depend on the specific deployment, here we describe an illustrative example.

As shown below, a three-level HSDN hierarchy is sufficient to scale the DC/DCI to tens of millions of servers.

With three levels, a possible design choice for the UP1s is to have each UP1 correspond to a DC. With this choice, the UP0 corresponds to the DCI and the UPBN1s are the DCGWs in each DC (the UPBG1s group the DCGWs in each DC).

Once the UP1s are chosen this way, a possible design choice for the UP2s is to have each UP2 correspond to a group of racks, where each group of racks may correspond to a portion of a row of racks, an entire row of racks, or multiple rows of racks. The specific best choice of how many racks should be in a group of racks corresponding to each UP2 ultimately depends on the specific connectivity, the number of servers per racks.

While precise numbers depend on the specific technologies used in each deployment, here and in the Scalability Analysis section below we want to give some ideas of the scaling capabilities of HSDN. For this purpose, we use some hypothetical yet reasonable numbers to characterize the partitioning design example.

Assume the following: a) 20 DCs connected via the DCI/WAN; b) 50 servers per rack; c) 20 racks per group of racks; d) 50 groups of racks per DC.

With these numbers, there are 500K servers per DC, for a total of 10M underlay network endpoints in the DC/DCI.

In the HSDN structure in this example, there are 20 UP1s, 500 UP2s per UP1, and 1000 servers per UP2.

3.3. MPLS-Based HSDN Forwarding

The hierarchically partitioned structure and the corresponding label stack are used in HSDN to scale the forwarding plane horizontally while using LFIBs of surprising small sizes in the network nodes.

As explained above, each label in the HSDN label stack is associated with one of the levels in the hierarchy and is used to forward to destinations in the underlay partitions at that level.

We describe the life of a packet in the HSDN DC/DCI. We use the specific design example described in Section 3.2.3 above to help in the explanation, but of course the forwarding would be similar for other design choices.

We first describe the behavior for non-TE traffic. In the HSDN DC/DCI, for a packet that needs to be forwarded to a specific endpoint in the underlay network, the outer label PL0 specifies which UP1 contains the endpoint. Let’s refer to this UP1 as UP1-a. For ECMP traffic, the PL0 binding is with a FEC corresponding to the UPBG1-a associated with UP1-a. Note that all the endpoints reachable via UP1-a are forwarded using the same FEC entry for Level 0 in the hierarchical partitioning.

Once the packet reaches one of the network nodes UPBN1-a in the UPBG1-a group (the upstream network nodes perform ECMP load balancing, thus the packet may enter UP1-a via any of the UPBN1-a nodes), the PL0 is popped and the PL1 is used for forwarding in the UP1-a. To be precise, because of penultimate hop popping, it is the network node immediately upstream of the chosen UPBN1-a that pops the label P0).

The PL1 is used within UP1-a to reach the UP2 which contains the endpoint. Let’s refer to this UP2 as UP2-a. In the UP2 network nodes the PL1 binding is with a FEC corresponding to the UPBG2-a associated with UP2-a. Similarly as above, note that all the endpoints reachable via UP2-a are forwarded using the same FEC entry for Level 1 in the hierarchical partitioning.

Once the packet reaches one of the network nodes UPBN2-a in the UPBG2-a group (once again, the upstream network nodes perform ECMP load balancing, so the packet may transit to any of the UPBN2-a nodes), the PL1 is popped and the PL2 is used for the rest of the forwarding (again, to be precise, the penultimate network node upstream of UPBN2-a is the one popping the PL1 label).

The PL2 is used within UP2-a to reach the desired endpoint. Note that the UPBN2 nodes and the network nodes in the UP2s have entries in their LFIBs only to reach endpoints within their UP2. They can reach endpoints in other UP2s by using a FEC entry corresponding to the UP2 containing the destination endpoint, identified by PL1.

The following two observations help in further clarifying the forwarding operation above.

- The PL0 is used for forwarding from the source to the UPBN1-a. For a packet originating from an endpoint attached to a certain UP2, say UP2-b, nested to a different UP1, say UP1-b, PL0 is used for forwarding in all network nodes that the packet transits until it reaches the UPBN1-a. This includes network nodes in UP2-b and UP1-b (i.e., "on the way out" from UP2). It also includes one of the UPBN1-b nodes. Note, however, that the PL0 is not popped at the UPBN1-b, since it is used for forwarding to the destination UPBN1-a.

- Not all packets carry a three-label MPLS stack. For example, a packet originating from the endpoint in UP2-b and destined to an endpoint in the same UP2-b only carries PL2. Similarly a packet originating from the endpoint in UP2-b and destined to an endpoint in a different UP2 nested in the same UP1-b only carries PL1 and PL2.

In the case of TE traffic, the use of the different labels in the label stack is similar as what described above for ECMP traffic. However, the labels are bound to FECs identifying a specific path within each UPs that is traversed. Therefore, TE traffic contributes for additional entries in the LFIBs in the network nodes. By properly designing the UPs, the number of LFIB entries can be kept relatively small.

HSDN, by superimposing a hierarchically-partitioned structure and using a label stack constructed according to such a structure, is able to impose a forwarding scheme that is aggregated by construction. This translates in dramatic reductions in the size of the LFIBs in the network nodes, since each node only needs to know a limited portion of the forwarding space.

HSDN supports any label assignment scheme to generate the labels in the label stack. However, if a label assignment scheme that is consistent with the HSDN structure is used, additional simplifications of the LFIBs and the control plane can be achieved.

In Section 5 below, we present one example of such a scheme, where the labels in the label stack represent the "physical" location of the endpoint, expressed according to the HSDN structure. For TE traffic, the labels represent a specific path towards the desired destination through the HSDN structure.

In the Scalability Analysis section and in the Control Plane section below we assume that such a Label Assignment scheme is used.

In HSDN, the LFIBs in the network nodes can be configured in such a way that all the paths in the DC/DCI are pre-established. This can be achieved using surprisingly small LFIB sizes.

4. Scalability Analysis

In this section, we compute the maximum size of the LFIBs for non-TE/ECMP traffic and any-to-any server-to-server TE traffic.

4.1. LFIB Sizing - ECMP

For ECMP traffic, at each level, all destinations belonging to the same partition at a lower level are forwarded using the same FEC entry in the LFIB, which identifies the destination UPBG for that level, or the destination endpoint at the lower level. Since the UPs are designed in such a way to keep the number of destinations small in all UPs, and the network nodes only need to know how to reach destinations in their own UP and in the adjacent UP at the higher level in the hierarchy, this translate to the fact that hyper scale of the DC/DCI can be achieved with very small LFIB sizes in the all individual network nodes.

The worst case for the LFIB size occurs at one of the network nodes that serve as UPBNs for one of the levels of UPs in the hierarchy. The level where the LFIB size occurs depend on the specific choice of the partitioning design.

To be completed.

4.2. LFIB Sizing - TE

As noted above, TE traffic may add a considerable number of entries to LFIB, since it creates one new FEC per TE tunnel to each destination.

HSDN provides a solution to this problem. In fact, HSDN can support any-to-any server-to-server "TE Max Case" with small LFIB sizes. In TE Max Cases, all sources are connected to all destinations (e.g., server to server) with TE tunnels, the tunnels using all possible distinct paths in the network. TE Max Case gives therefore an upper bound to the number of TE tunnels (and consequently, LFIB entries) in the network.

Again, in HSDN, since the UPs are designed in such a way to be relatively small, the number of paths in each partition can be kept to a manageable number.

In a Clos Topology (the analysis can be extended to generic topologies), the number of paths in a UP with N destination can be easily computed. The number of paths (and the maximum number of LFIB entries is equal to the products of the switch fanout in each tier traversed from the source to the destination in that UP. We refer to this as the TE Fanout Multiplicative Effect, which is illustrated in Figure 5.

Total # LFIB Entries for TE Max Case = N * F1 * F2 * ... * F(M-1)

Where Fi is the fanout of a switch in each tier traversed to the destination, M is the number of tiers in the UP, and N is the number of destinations in the UP.

Once again, by properly designing the UPs, the TE Fanout Multiplicative Effect can be kept under control, since the path computation is local for each of the UPs.

```
                                  +-------+
           Source                 |  Src  |
                                  | Node  |
                                  +-------+  F1=3
                                 /    |    \
                                /     |     \
                        +-------+ +-------+ +-------+
           Tier 1       |       | |       | |       |
                        |  Node | |  Node | |  Node |
                        +-+-----+ +---+---+ +-----+-+  F2=3
                          |   \ \ /   |   \ / /   |
                          |    \/ \---|---/ \/    |
                          |   / \ /---|---\ / \   |
                          | /  / \    |    / \  \ |
                        +-+-----+ +---+---+ +-----+-+
           Tier 2       |       | |       | |       |
                        |  Node | |  Node | |  Node |
                        +-------+ +-------+ +-------+  F3=2
                          \      \ /     \ /     /
                           \     / \     / \    /
                            \   /    / \    \  /
                             +-------+ +-------+
                             |       | |       |
           Tier 3            | Node  | | Node  |
                             +-------+ +-------+  F4=1
                                 \        /
                                  \      /
                                  +-------+
                                  | Dest  |
           Destination            | Node  |
                                  +-------+
```

Figure 5. Fan out multiplicative effect with TE.

To be completed.

5. HSDN Label Stack Assignment Scheme

HSDN can use any scheme to assign the labels in the label stack. However, if a label assignment scheme which assigns labels in a way consistent with the HSDN structure, important simplifications can be achieved in the control plane and in the LFIBs.

For non-TE FECs, the HSDN label assignment scheme assigns labels according to the "physical" location of the endpoint in the HSDN structure. Continuing our design example from above, for an endpoint X in UP2-a, PL0 would identify the DC in which the endpoint is located, PL1 would identify the group of racks in which the endpoint is located within the DC, and PL2 would identify the endpoint within the group of servers within the DC.

For TE FECs, the HSDN label assignment scheme assigns labels to identify a specific path in each UP that is traversed. In our example, for a specific TE tunnel to endpoint X, PL0 would identify the specific path that should be followed in the DCI, PL1 would identify the path that should be followed within the DC to reach the group of racks, and PL2 would identify the path to reach the endpoint within the group of racks (if there are multiple paths).

In order to assign labels to both non-TE traffic and TE traffic, HSDN uses a label format in which the labels are divided into two logical sub-fields, one identifying the destination within the UP, called Destination Identifier (DID), and one identifying the path, called Path Identifier (PID). The Path Identifier is only relevant for TE traffic, and can be zero for non-TE traffic. The HSDN Label format is illustrated in Figure 6.

```
0                           d                       19
+++++++++++++++++++++++++++++++++++++++++++++++++++++
|  Destination Identifier  |     Path Identifier    |
+++++++++++++++++++++++++++++++++++++++++++++++++++++
|<-------- d bits -------->|<-----(20-d bits)------>|
LSB                                               MSB
```

Figure 6. HSDN Label format.

Depending on the LFIB configuration, the two MSBs may be reserved for identifying the layer (i.e., whether the label is PL0, PL1, or PL2) to resolve ambiguity (not shown in Figure 6).

By properly designing the UPs, this label assignment scheme can support the desired scalability and the support of end-to-end TE traffic.

Note that by using this type of label assignment scheme important benefits can be achieved, including:

- The LFIBs become rather "static," since the FECs are tied to "physical" locations and paths, which change infrequently. This simplifies the use of the SDN approach to configure the LFIBs via a controller.

- All paths in each ECMP group use the same outgoing labels. This guarantees that a single LFIB entry can be used for each ECMP group.

The label stack needs to be imposed at the entry points. For an endpoint, this implies that the server NIC must be able to push a three-label stack of path labels (in addition to possibly one additional VL label for the overlay network).

6. HSDN Architecture - Control Plane

HSDN has been designed to support the controller-centric SDN approach in a scalable fashion. HSDN also supports the traditional distributed control plane approach.

HSDN introduces important simplifications in the control plane and in the network state as well.

6.1. The SDN approach

In the controller-centric SDN approach, the SDN controller configures the LFIBs in all the network nodes. With HSDN, the hierarchical partitioned structure offers a natural framework for a distributed implementation of the SDN controller, since the control plane in each UP is largely independent from other UPs.

For example, a possible architecture uses a SDN controller for each UP. Such SDN partition controller is in charge of configuring the LFIBs in the network nodes in the corresponding UP.

The SDN partition controller may also be in charge of TE computation. With proper design of the UPs, TE path computation algorithms which perform partition-local computation while approach global optimality can be used.

To be completed.

6.2. Distributed control plane

HSDN can also use the traditional distributed routing protocol approach to distribute HSDN labels, for example using BGP [RFC3107].

To be completed.

7. Security Considerations

When the SDN approach is used, the protocols used to configure the LFIBs in the network nodes MUST be mutually authenticated.

For general MPLS/GMPLS security considerations, refer to [RFC5920].

Given the potentially very large scale and the dynamic nature in the cloud/DC environment, the choice of key management mechanisms need to be further studied.

To be completed.

8. IANA Considerations

TBD.

9. References

9.1 Normative References

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

[RFC3031] Rosen, E., Viswanathan, A., and R. Callon, "Multiprotocol Label Switching Architecture", RFC 3031, January 2001.

[RFC3032] Rosen, E., Tappan, D., Fedorkow, G., Rekhter, Y., Farinacci, D., Li, T., and A. Conta, "MPLS Label Stack Encoding", RFC 3032, January 2001.

[RFC3107] Rekhter, Y. and E. Rosen, "Carrying Label Information in BGP-4", RFC 3107, May 2001.

9.2 Informative References

[RFC5920] Fang, L., Ed., "Security Framework for MPLS and GMPLS Networks", RFC 5920, July 2010.

Authors' Addresses

Luyuan Fang
Microsoft
5600 148th Ave NE
Redmond, WA 98052
Email: lufang@microsoft.com

Vijay Gill
Microsoft
5600 148th Ave NE
Redmond, WA 98052
Email: vgill@microsoft.com

Fabio Chiussi
Seattle, WA
Email: fabiochiussi@gmail.com