INTERNET-DRAFT
Intended Status: Informational
Expires: July 26, 2016

Luyuan Fang
Deepak Bansal
Microsoft
Fabio Chiussi

Chandra Ramachandran
Juniper Networks
Ebben Aries
Facebook
Shahram Davari
Broadcom
Barak Gafni
Mellanox
Daniel Voyer
Bell Canada
Nabil Bitar
Verizon

January 23, 2016

MPLS-Based Hierarchical SDN for Hyper-Scale DC/Cloud
draft-fang-mpls-hsdn-for-hsdc-05

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 Equal Cost Multi Path (ECMP) load-balanced traffic and any-to-any end-to-end Traffic Engineered (TE) traffic. HSDN achieves massive scale using surprisingly small forwarding tables in the network nodes. HSDN introduces a new paradigm for both forwarding and control planes, in that all paths in the network are pre-established in the forwarding tables and the labels can identify entire paths rather than simply destinations. 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 endpoints and paths according to the HSDN structure; and 4. Forwarding using the HSDN MPLS labels.

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering

Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/1id-abstracts.html

The list of Internet-Draft Shadow Directories can be accessed at 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 both Equal-Cost Multi-Path (ECMP) load-balanced traffic and any-to-any Traffic Engineering (TE) traffic.

Being able to scale at low cost requires to avoid 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 in order to support Virtual Machine (VM) and Virtualized Network Function (VNF) mobility.

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, since a single destination may contribute multiple entries in the tables.

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, since each TE tunnel corresponds to an entry in the tables.

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 accepted as "the way to build the next generation clouds," still needs to be demonstrated to be scalable to the levels required in the hyper-scale DC and cloud.

Finally, the underlay network architecture should offer certain capabilities to facilitate the support of the demands 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 with any forwarding technology, including IPv4 and IPv6, it has been designed to leverage Multi Protocol Label Switching (MPLS)-based forwarding [RFC3031], using label stacks [RFC3032] constructed according to the HSDN structure. This document therefore describes MPLS-based HSDN. Here, we describe end-to-end (host-to-host) MPLS-based HSDN, where the entire HSDN label stacks from source to destination are imposed at the server's Network Interface Cards (NICs), and thus all the IP lookups are confined to the network edges. However, MPLS-based HSDN does not need to be end-to-end, since label imposition could happen instead at the network nodes (e.g., at the Top-of-Rack (ToR) switches), or intermediate lookups in the network could be introduced, or even a combination of MPLS and IP forwarding could be deployed as part of the HSDN network.

The HSDN underlay network is suited to support any Layer 2 or Layer 3 virtualized overlay network technology. In this document, we assume a MPLS-based overlay technology using a Virtual Network (VN) Label, which is encapsulated in the HSDN label stack. However the description can be easily generalized to any overlay technology, such as BGP/MPLS IP VPNs [RFC4364], EVPN [RFC7432], VXLAN [RFC7348], NVGRE [RFC7637], Geneve [I-D.draft-gross-geneve], and other technologies.

HSDN achieves massive scale using surprisingly small LFIBs in the network nodes, while supporting both ECMP load-balanced traffic and any-to-any end-to-end TE traffic [HSDNSOSR15]. HSDN also brings important simplifications in the control plane and in the architecture of the SDN controller.

The HSDN architecture and operation is characterized by two fundamental properties. First, all paths in the network are pre-

established in the forwarding tables. Second, the HSDN labels can identify entire paths or groups of paths rather than simply destinations.

These two properties radically simplify establishing and handling tunnels. In addition to optimally handling both ECMP and Non-Equal Cost Multi Path load balancing, HSDN enables any-to-any, end-to-end, server-to-server TE at scale. With HSDN, the "cost" of establishing a tunnel is essentially eliminated, since the "tunnels" are pre-established in the network, and the TE task becomes one of path assignment and bandwidth allocation to the flows. As a larger portion of the traffic can be engineered effectively, the network can be run at a higher utilization using comparatively smaller buffers at the nodes.

The HSDN forwarding architecture in the underlay network 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 control and forwarding, and have the LFIBs configured by a controller according to a full SDN approach. The controller-centric approach is described in this document. In this context, "MPLS forwarding" in HSDN simply means using MPLS labels to forward the packets, since there is no need for label distribution protocols.

However, the HSDN control plane can also be built using a hybrid approach, in which a routing or label distribution protocol is used to distribute the labels, in conjunction with a SDN controller. This hybrid approach may be particularly useful during technology migration. The use of BGP Labeled Unicast (BGP-LU) for label distribution and LFIB configuration in a HSDN architecture is described in [I-D.fang-idr-bgplu-for-hsdn].

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
BGP-LU	Border Gateway Protocol Labeled Unicast
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
NIC	Network Interface Card
PID	Path Identifier
SDN	Software Defined Network
SN	Spine Node
SVR	Server
UP	Underlay Partition
UPBG	Underlay Partition Border Group
UPBN	Underlay Partition Border Node
TE	Traffic Engineering
ToR	Top-of-Rack switch
TR	Top-of-Rack switch (used in figures)
VN	Virtual Network
VM	Virtual Machine
VNF	Virtualized Network Function
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 HSDN 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 Clos-based topology.

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

the overlay network traffic.

- 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.

- 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.

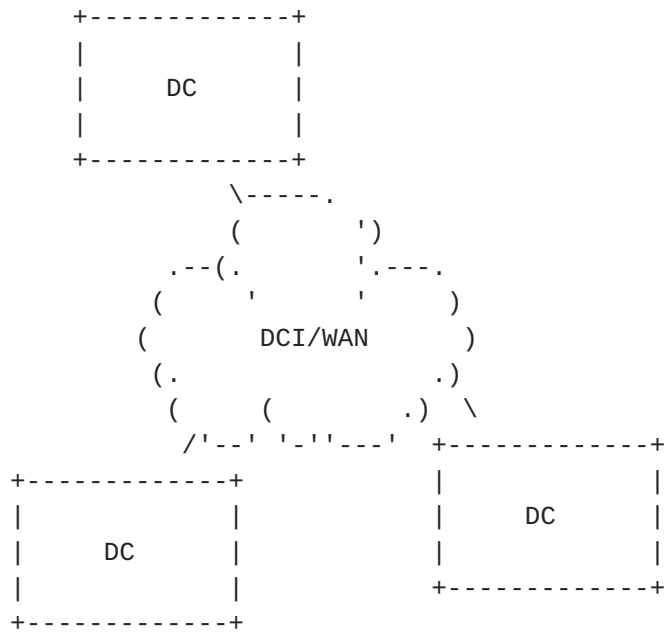

Figure 1. DCIWAN interconnecting multiple DCs.

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

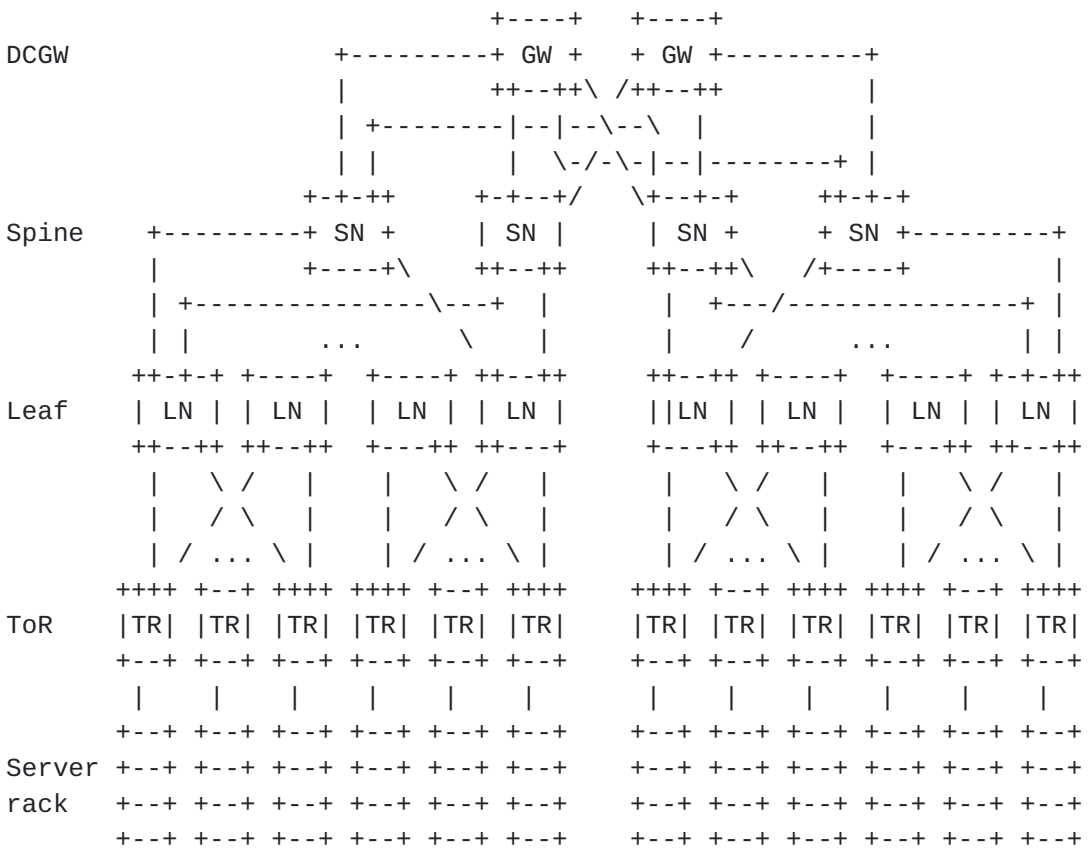

Figure 2. Typical Clos-based DC fabric topology.

Note: Not all nodes and 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 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.

Figure 2 is not meant to represent the precise topology of the DC. In fact, 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 leafs) or asymmetric topologies, not shown in the figure, may be used. A precise description of the possible topologies and related design criteria is out of the scope of this

document.

What is 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 tier (i.e., not all network nodes in a tier are necessarily connected to all 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 (in other words, in a large, fat Clos there are too many network nodes in some tiers for all network nodes to connect to one another), and to the physical constraints of the DC (i.e., the network nodes may be located physically apart in separate rooms or buildings, and full connectivity may become too costly).

In a typical DC, the racks of servers are physically organized in "clusters" of racks, and dedicated banks of leaf switches may serve the ToRs in each cluster. For example, the racks may be physically placed in rows of racks, and a cluster of racks may correspond to a portion of a row, an entire row, or multiple rows of racks. Indeed, the leaf nodes are sometimes called "middle (or end) of the row switches" because they are physically located in a rack in the middle (or end) of a row of racks of servers. In turn, leaf nodes may also be organized in "zones" (we use "clusters" and "zones" as generic terms, but other terms may be used in the industry to refer to similar concepts), and banks of spines may be assigned to serve each zone. For example, a zone may include all the banks of leaf nodes that are in a room or in a building in the DC.

The actual connectivity is typically organized following an aggregation/multiplexing connectivity architecture that consolidates traffic from the edges into the leafs and spines, while allowing for over-subscription in order to strike a reasonable trade-off between cost and available capacity. The connectivity between each tier may use some form of shuffle-exchange topology that attempts to "mix" the available paths while taking in account the physical constraints.

The key observation is that it is impractical, uneconomical, and ultimately unnecessary to use a fully connected Clos-based topology in a large scale DC. Because of the physical constraints, the topology of a large DC is not a flat, fully-connected Clos, but rather has a dertain hierarchy. The HSDN architecture recognizes this fact, and uses it to dramatically simplify forwarding and control planes using an approach that is also hierarchical.

2. Requirements

2.1. MPLS-Based HSDN Design Requirements

The following are the key design requirements for MPLS-based 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 load-balanced traffic 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 full SDN controller approach, and 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 load balancing 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). For example, if the hash function used for ECMP forwarding is based on the IP 5-tuple, as is often the case, this requirement implies that the network nodes MUST be able to lookup the 5-tuple inside up to four labels.

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.

As explained in Section 3.3.1, the HSDN label stacks can be used to identify entire paths to each endpoint, rather than simply the destination endpoint itself. As a matter of fact, the HSDN solution is meant to be configured with all possible paths in the network pre-established in the LFIBs in the network nodes. In this case, a FEC per path to each endpoint is defined. However, because of the way the HSDN architecture is designed, the required local number of entries in the LFIB of each network node remains surprisingly small.

In this section, we explain in detail each of these concepts. Scalability analysis for both ECMP load-balanced and TE traffic 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.

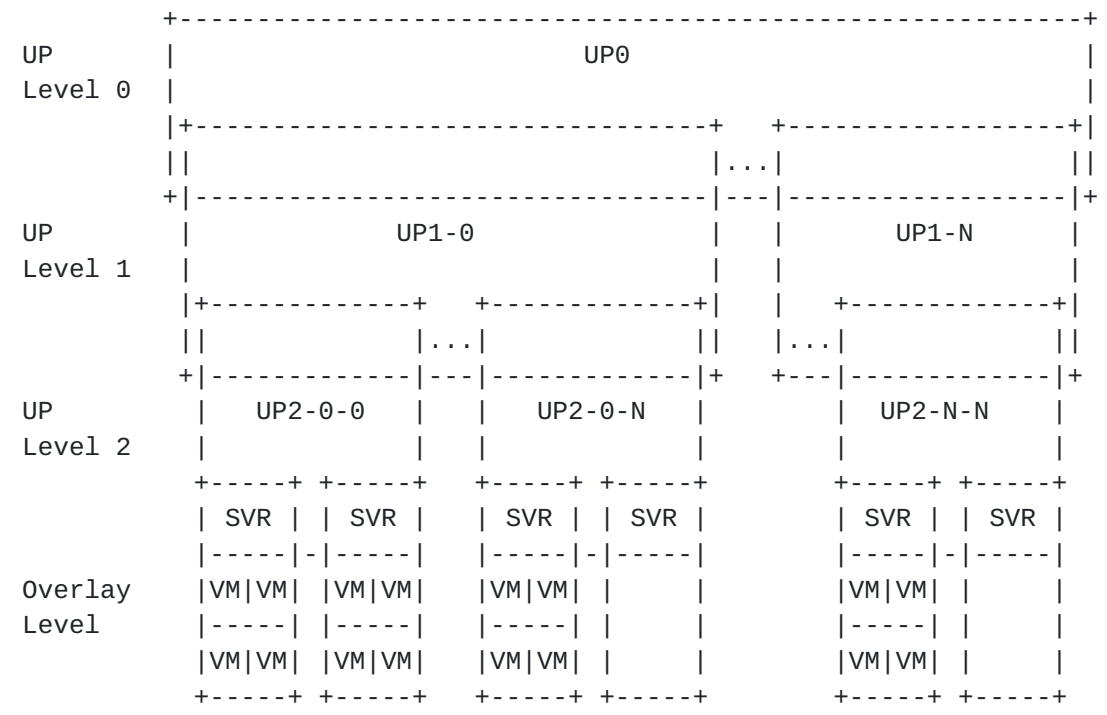

Figure 3. HSDN underlay network hierarchical partitioning of DC/DCI.

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 in Section 4, 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 nesting into each higher level partition (e.g., partition UP1-1 has M1 UP2 partitions, partition UP1-2 has M2 UP2 partitions, and so on).

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. As mentioned in the Introduction, the HSDN underlay network is suited to support any Layer 2 or Layer 3 virtualized overlay network technology, such as BGP/MPLS IP VPNs [RFC4364], EVPN [RFC7432], VXLAN [RFC7348], NVGRE [RFC7637], Geneve [I-D.draft-gross-geneve], and other technologies. A full description of the encapsulation of these technologies into the HSDN underlay label stack is out of scope of this document and will be addressed in a separate 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 level is that they need to follow the "natural" connectivity implemented in the DC/DCI topology. An example is given in Section 3.2.3 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.

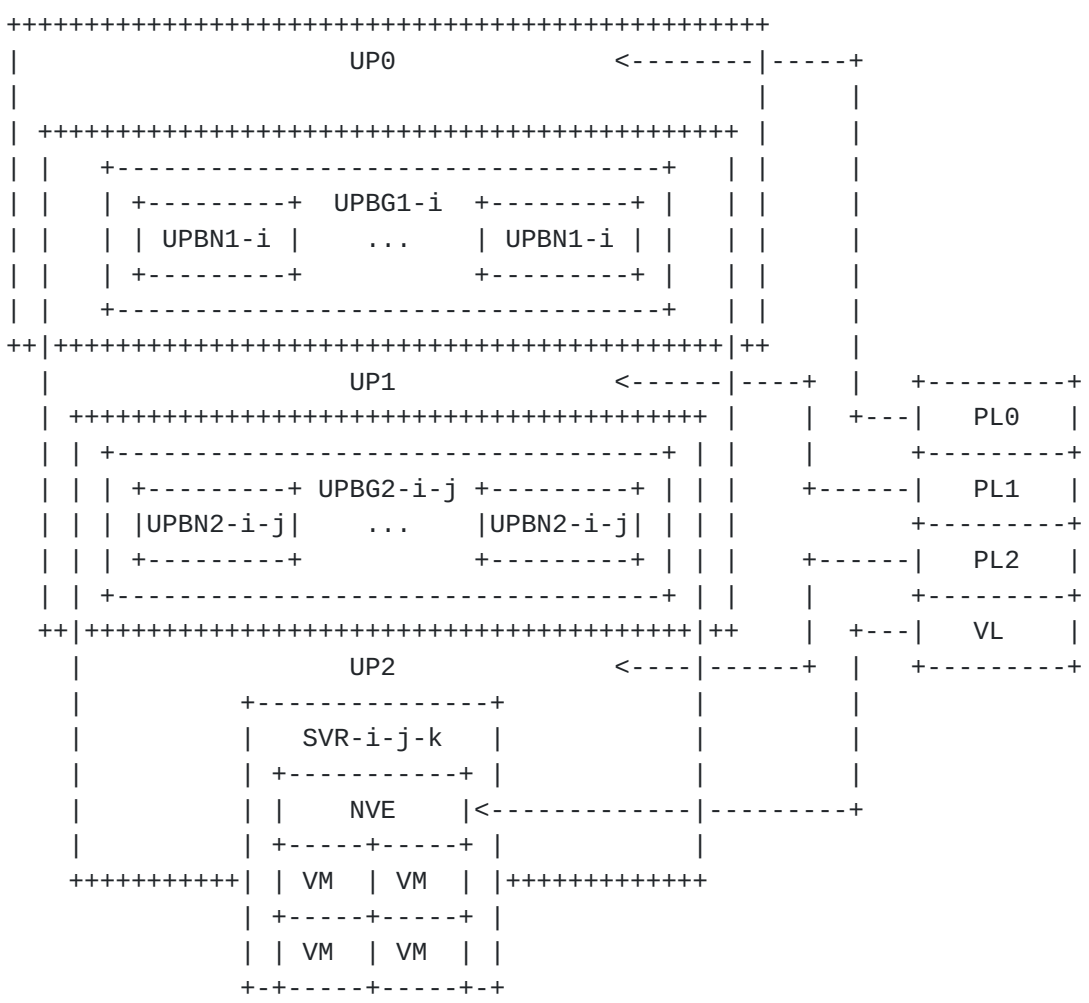

Figure 4. UBPNs, UBPGs, and label stack assignment.

The UPBNs serve as the connecting nodes between adjacent partitions. As such, 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 (and vice-versa, they are the exit points for traffic from a lower level partition to a higher 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 in Section 5, 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. As mentioned above, the connectivity of any large-scale DC is not fully flat, but rather contains some sort of hierarchical organization. Recognizing the hierarchy of the physical connectivity is an important starting point in the design of the partitions.

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, in addition to the UPBNs, 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 in Section 3.2.3.

As explained in more detail in Section 3.3, 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 (i.e., their own corresponding lower-level partition and the higher-level 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.

Similarly, in order to establish all possible paths in the entire network, the UPBNs need to have entries in their LFIBs only for all possible paths to the destinations in the two partitions to which they belong to.

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. Although finding the optimal design choice may require a little trial-and-error computation of different options,fortunately, for most practical deployments, it is relatively simple to find a good tradeoff that achieves the desired scalability

to millions or tens of millions of endpoints.

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 (for example, using an additional sequential number within the UPBG), which for simplicity is not shown here. This more granular naming convention is needed to configure the paths and the TE tunnels.

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 partitioning 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 destinations 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. As mentioned above, the HSDN underlay network is suited to support any Layer 2 or Layer 3 virtualized overlay network technology, and thus the VL may be a label, a tag, or some other identifier, depending on the overlay technology used. The details of the VL encapsulation and processing for different overlay technologies 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 UP2 to which the endpoint is attached to, and the PL2 specifies the FEC in the UP2 corresponding to the endpoint.

The labels in the HSDN label stack can identify entire paths, rather than simply the end destination within the corresponding partition. This can be used to bring dramatic simplifications in handling tunnels and TE traffic in particular, as further explained in Section 3.3.2.

As mentioned above, in this draft we describe end-to-end MPLS-based HSDN forwarding, where the entire HSDN label stacks from the sources to the destinations are inserted at the server's NICs. In this scenario, the label stack imposed at a server points all the way to the end destination of the packet, which may be in a different DC. With any-to-any, end-to-end TE, the HSDN label stack identifies the entire path to the destination. For inter-DC traffic, there may be cases where the path through the remote DC would be preferably determined when the packet arrives at that DC, or when the packet leaves the source DC, so it may be desirable that part of the label stack be imposed inside the network rather than at the server. Nothing precludes this design choice, and a lookup may be added where desired in the HSDN network.

A scheme to assign the PL labels in the HSDN label stack is described in Section 5.

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 (Section 4) 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 in Section 4.2), 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 conflicting objectives. Although the optimal design choices ultimately depend on the specific deployment, fortunately, it is generally rather straightforward to identify design choices that can support scalability to millions or tens of millions of servers.

Here we describe a design example to illustrate that 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 in the DC and 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

(Section 4) 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.

With HSDN, by superimposing a hierarchically-partitioned structure and using a label stack constructed according to such a structure, we are 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 (Section 4) and in the Control Plane section (Section 6) we assume that such a Label Assignment scheme is used.

In the rest of this section, 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.

3.3.1 Non-TE Traffic

We first describe the behavior for ECMP load-balanced, 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 up" from UP2). It also includes one of the UPBN1-b nodes.

It is important to note, however, that the PL0 is not popped at the UPBN1-b, since it is used for forwarding to the destination UPBN1-a.

- It should be pointed out that an important requirement for HSDN is to achieve route optimization for ECMP traffic, meaning that the hierarchy should forward a packet from any source to any destination using the same number of hops and without introducing any additional latency compared to a flat architecture. For example, a packet originating from an endpoint in UP2,N,M, and destined to an endpoint in the same UP2,N,M should not be forwarded all the way to the highest level in the hierarchy and back, but should be forwarded to the desired endpoint by "turning it around" towards the destination at the first node in the UP2,N,M that contains an entry to that desired endpoint. Indeed, if the packet turns around at the proper node, it will go through the same number of hops as it would have gone through in a flat architecture. This should hold true even in the case where the UP1s and/or UP2s contain intermediate tiers of switches and the packet needs to be turned around in the intermediate nodes.

  Route optimization is easily achieved in HSDN by simply having the packets only carry the portion of the label stack that is needed to reach the destination using the appropriate turn around node. Continuing with our example, the packet above only needs PL2 to be optimally forwarded, since it should never "go out" of UP2,N,M. Thus, PL0 and PL1 should not be included in its label stack, to avoid an unnecessary round trip up and down the DC through all the levels in the hierarchy. Similarly, a packet originating and terminating in the same UP1, but in different UP2s, only needs PL1 and PL2 to be forwarded.

  In this case, a network node would have to process different labels for traffic going up and out the partition versus traffic staying in the partition ("going up" and "coming down" refer to the direction of traffic in Figure 3). Since the label spaces for the two path labels may overlap, ambiguity would result. Depending on the LFIB configuration, the two Most Significant Bits (MSBs) in each label in Figure 4 may be reserved for identifying the layer (i.e., whether the label is PL0, PL1, or PL2) and resolve ambiguity.

  A better solution to achieve the same without using precious bits

in the labels is to use a "turn around entry" in the LFIBs, which flags that the packet needs to turn around at that node and the relevant label is not the outer label (as it would be for traffic going up or coming down, for which the outer label just needs to pass through), but is the one underneath (thus, the outer label needs to be popped to expose the relevant label). In our example, the packet destined to an endpoint in the same UP2,N,M of the originating server may carry a PL1 corresponding to the "turn around" label value and a PL2 corresponding to the desired endpoint within UP2, and does not need a PL0.

In the case of ECMP load-balanced non-TE traffic, the labels in the HSDN label stack identify ECMP groups for each destination in the corresponding partition. In this way, at each node in the partition, the outgoing label is the same for all paths belonging to the same ECMP group. A label allocation scheme for this is described in Section 5.

3.3.2 TE Traffic

Handling TE traffic in the hyper-scale DC/DCI presents major scalability challenges, since each TE tunnel contributes one entry in the forwarding tables, and the TE path and bandwidth allocation computation is a NP-complete problem.

HSDN introduces radical simplifications in establishing and handling tunnels, and in supporting TE in particular.

In HSDN, all paths in the network can be pre-established in the LFIBs. Because of the way the HSDN architecture is constructed, the number of entries that have to be stored in the local LFIB in each network node remains surprisingly small.

In this case, the labels in the HSDN stack identify entire paths, or groups of paths, to each destination in each partition, rather than just the destination itself.

With HSDN, since the "cost" of establishing a tunnel is essentially eliminated (all "tunnels" are pre-established in the network), and the TE task becomes one of path assignment and bandwidth allocation to the flows. Furthermore, the hierarchical structure of HSDN makes it possible to devise algorithms and heuristics for path and bandwidth allocation computation that operate largely independently in each partition, and are therefore computationally feasible even at large scale. A description of such algorithms is out of scope of this document. As a larger portion of the traffic can be engineered effectively, the network can be run at a higher utilization using comparatively smaller buffers at the nodes.

Since all paths can be accommodated in the LFIBs, HSDN makes it possible to support "TE Max Case" with small LFIB sizes. In TE Max Case, 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.

The fact that the LFIBs remain relatively small even when all possible paths are configured is the consequence of two desirable properties of HSDN.

First, since in HSDN the individual 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.

Second, the hierarchical structure of HSDN makes it possible to use the partitioning astutely to break the "TE Fanout Multiplicative Effect," which defines the number of paths to a destination, and can easily contribute to the LFIB explosion as the number of hops and the fanout of each hop to each destination in the network increases. As explained in Section 4.2, with the hierarchical structure, the TE Fanout Multiplicative Effect is only multiplicative within each level in the hierarchy. Thus, by properly designing the partitioning, the multiplicative effect can be kept to a manageable level.

In the case of TE traffic, the processing of the different labels in the label stack is similar to what described above for ECMP load-balanced non-TE traffic. However, the labels are bound to FECs identifying a specific path within each UPs that is traversed.

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 all the individual network nodes.

A detailed explanation of how the LFIB size can be computed in all the nodes of an HSDN network is given in [HSDNSOSR15]. 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.

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, and in fact, HSDN can support any-to-any server-to-server "TE Max Case" with small LFIB sizes.

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. This is the "TE Fanout Multiplicative Effect" mentioned above, which is illustrated in Figure 5. Accordingly:

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. HSDN breaks the multiplicative effect, since the TE Fanout Multiplicative Effect is multiplicative only within each UP, rather than in the entire network, and the "multiplication" restarts at each level of the hierarchy. A detailed description of the LFIB computation in all network nodes to support TE Max Case is given in [HSDNSOSR15].

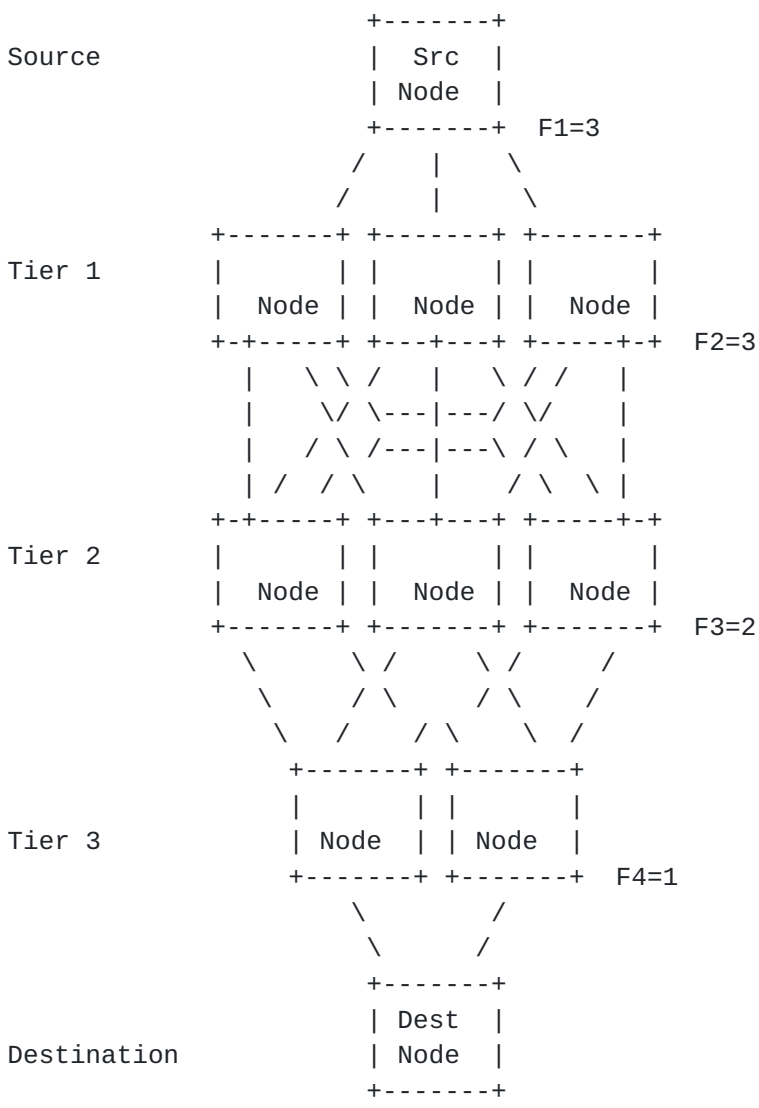

Figure 5. Fan out multiplicative effect with TE.

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.

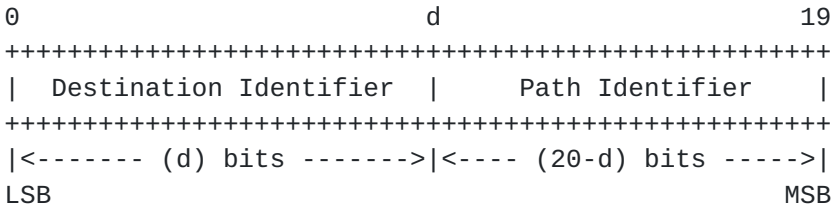

Figure 6. HSDN Label format.

In this label assignment scheme, the path labels associated with a partition are globally unique within that partition, meaning that different partitions at the same level can use the same label space. For PL0, the path labels are globally unique within the entire network, since there is only one UP0. Neither of these is a scaling limitation, since all partitions are relatively small.

The bits in the DID for each level must be sufficient to represent the distinct destinations that need to be known in the UPs at that level, and the bits in the PID for each level must be sufficient to represent all the distinct paths that need to be defined in the UPs at that level (closed-form expressions for both these numbers are given in [HSDNSOSR15]). In practice, this is not a significant scalability constraint: with three MPLS labels in the stack, partitioning architectures and label formats according to this scheme can be found to scale to tens of millions of servers.

Depending on the LFIB configuration, the two MSBs may be reserved for

identifying the level (i.e., whether the label is PL0, PL1, or PL2) to resolve ambiguity (not shown in Figure 6). Note, however, that this is not strictly necessary and the same function of identifying the level can be achieved by simply allocating "turn around" entries in the nodes, as explained in Section 3.3.1, so an individual node always sees the same label in the stack.

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 push 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. The individual UP control planes operate in parallel, with loose synchronization among one another.

Therefore, the HSDN control plane is logically partitioned in a way

that is consistent with the forwarding plane partitioning. Each UP is assigned a corresponding UP controller, which configures the LFIBs in the network nodes in the corresponding UP. The individual UP controllers communicate with one another to exchange the labels and construct the label stacks. In HSDN, configuring the LFIBs in the network nodes is not a difficult task, since the labels are static and configuration updates are needed only when the physical topology changes or endpoints are added or permanently removed, and thus they are not too frequent.

Each UP controller at the lowest level of the hierarchy is also in charge of providing the label stacks to the server's NICs in the corresponding partition. For this purpose, a number of label servers, which may also be arranged in a hierarchy, are used to provide the mappings between IP addresses and label stacks.

Redundancy is superimposed to the structure of UP controllers, with each UP controller shadowing UP controllers in other UPs.

The HSDN UP controllers may also be in charge of TE computation. HSDN TE path computation algorithms that perform for the most part partition-local computation (so the computation is also horizontally scalable) but still approach global optimality using inter-UP-controller synchronization at a different time scale, can be devised.

6.2. HSDN Distributed Control Plane

The HSDN control plane can also be built using a hybrid approach, in which a routing or label distribution protocol is used to distribute the labels, in conjunction with a controller. An example using BGP-LU [RFC3107] is presented in [I-D.fang-idr-bgplu-for-hsdn].

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. Acknowledgments

We would like to acknowledge Yakov Rekhter for many discussions related to HSDN.

10. Contributors

Vijay Gill
Salesforce
Email: vgill@salesforce.com

Linda Dunbar
Huawei Technologies
5430 Legacy Drive, Suite #175
Plano, TX 75024
Email: linda.dunbar@huawei.com

Andrew Qu
MediaTek
2860 Junction Ave.
San Jose, CA 95134
Email: andrew.qu@mediatek.com

Jeff Tantsura
Ericsson
200 Holger Way
San Jose, CA 95134
Email: jeff.tantsura@ericsson.com

Wen Wang
Century Link
2355 Dulles Corner Blvd.
Herndon, VA 20171
Email: wen.wang@centurylink.com

Himanshu Shah
Ciena
3939 North 1st Street
San Jose, CA 95112
Email: hshah@ciena.com

Ramki Krishnan
Dell
Email: ramki_krishnan@dell.com

Iftekhar Hussain
Infinera Corporation
140 Caspian Ct,

Sunnyvale, CA 94089
Email: ihussain@infinera.com

11. References

11.1 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.

[RFC4364] Rosen, E. and Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4364, February 2006.

[RFC7432] Sajassi et al., "BGP MPLS Based Ethernet VPN", RFC 7432, February 2015.

11.2 Informative References

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

[RFC7348] M. Mahalingam et al., "Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks", RFC 7348, August 2014.

[RFC7637] P. Garg et al., "NVGRE: Network Virtualization using Generic Routing Encapsulation", RFC 7637, Sept. 2015.

[I-D.fang-idr-bgplu-for-hsdn] L. Fang et al., "BGP-LU for HSDN Label Distribution", draft-fang-idr-bgplu-for-hsdn-02 (work in progress), July 2015.

[I-D.draft-gross-geneve] J. Gross et al., "Geneve: Generic Network Virtualization Encapsulation", draft-gross-geneve-02 (work in progress), October 2014.

[HSDNSOSR15] L. Fang et al., "Hierarchical SDN for the Hyper-Scale,

Hyper-Elastic Data Center and Cloud", ACM SIGCOMM Symposium on SDN Research 2015, Santa Clara, CA, June 2015.

Authors' Addresses

Luyuan Fang
Microsoft
15590 NE 31st St.
Redmond, WA 98052
Email: lufang@microsoft.com

Deepak Bansal
Microsoft
15590 NE 31st St.
Redmond, WA 98052
Email: dbansal@microsoft.com

Fabio Chiussi
Seattle, WA 98116
Email: fabiochiussi@gmail.com

Chandra Ramachandran
Juniper Networks
Electra, Exora Business Park Marathahalli - Sarjapur Outer Ring Road
Bangalore, KA 560103, India
Email: csekar@juniper.net

Ebben Aries
Facebook
1601 Willow Road
Menlo Park, CA 94025
Email: exa@fb.com

Shahram Davari
Broadcom
3151 Zanker Road
San Jose, CA 95134
Email: davari@broadcom.com

Barak Gafni
Mellanox
6 Habarzel St.
Tel Aviv, Israel
Email: gbarak@mellanox.com

Daniel Voyer

Bell Canada
Email: daniel.voyer@bell.ca

Nabil Bitar
Verizon
40 Sylvan Road
Waltham, MA 02145
Email: nabil.bitar@verizon.com