This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2009 proceedings

An Evolutionary Approach to End-to-End Addressing and Routing in *all-Ethernet* Wide-Area Networks

*Ashwin Gumaste, ++Mohit Chamania, and ++Admela Jukan
*RLE, EECS, Massachusetts Institute of Technology, Cambridge, USA, Dept of CSE, Indian Institute of Technology, Bombay
++Technische Universität Carolo-Wilhelmina zu Braunschweig, Germany
ashwing@ieee.org, {chamania, jukan}@ida.ing.tu-bs.de

***Abstract*- While the introduction of new Ethernet-based wide-area solutions, such as *provider-backbone bridging traffic engineering*– PBB-TE, paves a way for Ethernet to become a carrier class service, it is restricted to the metro area and hence unable to provision global end-to-end communication. In this paper, we propose a new scheme for *all-Ethernet wide area* networking that involves a unique addressing and routing mechanism, and leads to a scalable, hierarchical and service-oriented transport network architecture. The scalability is achieved by means of abstraction of any irregular physical topology into a regular logical topology, based on the concept of *binary trees*. The regular logical topology is represented with logical 1x2 Ethernet switches as the fundamental building blocks which allow switching using a unique binary addresses in a simple and automated fashion. We propose an evolutionary architecture to provide end-to-end Ethernet routes using binary addresses embedded in stacked VLAN tags on native Ethernet frames, in line with the emerging standards. The results show that a significant simplification of wide-area inter-networking can be achieved, while supporting carrier-grade network performance with all-Ethernet features.**

***Keywords* – Carrier Ethernet, Addressing, Routing, Binary Tree.**

I. INTRODUCTION

Ethernet has evolved from a simple, adaptable LAN technology to a resilient carrier-class wide-area networking technology over a period of three decades. Universal adoption of Ethernet interfaces by providers, customers and enterprises has made this technology the dominant choice for inter-networking. The salience of Ethernet is in its simplicity, flexibility and ability to provide low-cost, multi-protocol and ubiquitous communication. Originally introduced as a LAN technology, with carrier sensing, Ethernet has undergone a significant transformation now appearing as the dominant transport choice in provider WANs (metro/long-haul). Today, the so-called *Carrier-grade Ethernet* is incorporating features of WANs, such as scalability, resiliency, fault management (operations and administration), and predictable QoS [1]. Recently, specific standards have emerged to extend the reach of native Ethernet services by modifying appended VLAN technology such as Q-in-Q [2], encoding MAC in MAC together which have led to Provider Backbone Bridging-Traffic Engineering (PBB-TE) and T-MPLS [3][4].

While the introduction of new Ethernet based wide-area solutions makes it possible for Ethernet to offer a carrier class service, it uses a global control plane and assumes a flat networking hierarchy and address space, thereby making it difficult to scale to encompass the end-to-end aspects of communications. This is because flat addressing in Ethernet requires large lookups at every node and while it is an attractive proposition for isolated islands of network domains, it is unlikely to scale to form a global end-to-end Ethernet system. In other words, if the current carrier grade Ethernet is to become a new global network paradigm, the addressing scheme should encompass the present hierarchy of networks, ranging from access, over metro to the core network, while maintaining its simplicity, cost-efficiency, and easy upgradeability.

In this paper, we propose a simple, scalable and service friendly global all-Ethernet architecture built upon a novel addressing and routing approach. Our proposal is based on the following three key concepts. First, an irregular physical topology is mapped onto a regular logical topology based on binary trees. Second, the regular logical topology is then represented as a collection of several lumped 1x2 switches. Third, by using *binary* routing, these switches facilitate automatic frame forwarding using binary addresses that are assigned to every end-node embedded in Ethernet frames. Our approach is evolutionary, as we can soft upgrade existing IP/Ethernet/optical infrastructure. At the same time, it is scalable, implying that nodes can be added into the network without the explicit reconfigurations of the entire network.

The paper is organized as follows. In Section II, we describe the proposed architecture and the abstraction of the irregular hierarchical physical topology to a regular logical topology, based on binary trees. In Section III, we discuss the implementation of the proposed addressing and routing mechanism as well as end-to-end communications in all-Ethernet networks. In Section IV, we discuss related work. Section V presents numerical results validating our proposal. Finally, Section VI concludes the paper.

II. THE ALL-ETHERNET ARCHITECTURE

The present global network hierarchy can be abstracted through a 3-tier model with elements in each tier having specific characteristics as shown in Fig. 1. The 1st tier consists of few, large core nodes, connected in a physical mesh. This is the core of the network often termed as the long-haul. The 2nd tier resembles a tree; the tree is a virtual tree and several such trees are built together to form a virtual (logical) mesh, often overlaid on a physical ring. The root of this tree is at one or more of the core nodes while its leaves terminate at a service providers' Point of Presence (POP). The 2nd tier is what is popularly called the metro. The 3rd tier represents end-users connected to POPs via a multitude of physical media (wireless, optical, DSL). This tier is the access network which both physically

978-1-4244-3435-0/09/$25.00 ©2009 IEEE

and logically is a star with an aim to maximize the access line-card utilization of a provider's POP.

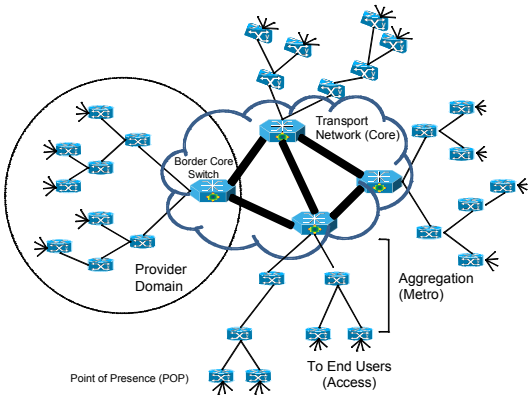

Fig. 1 The 3-tier hierarchy in today's networks.

From a routing perspective, the 1st tier (a collection of core-nodes) implies inter-provider routing in a mesh graph. The 2nd tier is a collection of nodes connected in a tree topology, with the characteristic *that the tree branches out rapidly* within the domain of a provider; the root of this tree is at one of the core nodes (see the large gateway node within the dashed circle in Fig. 1). Finally, the 3rd tier consists of access nodes connected at the ends of a star whose center node is the leaf of the metro tree topology. From this description, the varying topology and scale of the three tiers of the network coupled with the flat MAC addressing of Ethernet frames naturally leads to difficulty in developing efficient, scalable end-to-end Ethernet paths. Moreover, each tier desires different networking performance, characterized by delay, bandwidth and reliability.

Given the three tiers and associated diversity, the question is whether it is possible to create scalable end-to-end all-Ethernet provisioning and if so, what the requirements of such a system would be? To this end, we propose a new architecture which uses a *binary* addressing scheme. In every tier, the address is defined as a binary string used within the logical overlaid topology; we propose in fact a composite binary string, to reflect the three-tier hierarchy.

The main idea is to append binary strings to Ethernet frames, where bits in the frame are used to identify routing decisions at intermediate nodes. More precisely, bits govern the switch configuration at each node, which in turn explicitly determine the route. Recall that regular logical topologies (binary tree) are overlaid on physical topologies to facilitate routing; within that logical topology precisely, an end-to-end route is described by a binary string.

Physical topologies can be converted to tree topologies using an algorithm that we propose which enables creation of a binary tree in $O(N^2)$ time-complexity. In a binary tree, each node can be connected to a maximum of three other nodes. A node in the binary tree can be logically represented by 1x2 switches as shown in Fig 2. A packet at any incoming port of the binary node can be routed to only one of two possible output ports. Therefore, a single bit is used to define the route to be selected at a node in the binary tree. This implies that a binary string can be used to describe a route with the number of elements in the string corresponding to the number of switches from the source to the destination and each element corresponding to the state of an intermediate switch. As we traverse from a source to a destination node, we check each route string element, i.e., corresponding to the state of a particular switch. If the (element) value is *0*, then we direct the packet left-wards. If however, the value is 1, the packet is directed right-wards.

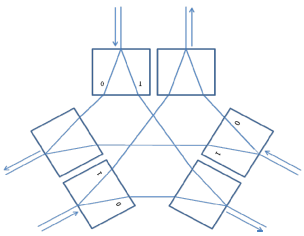

Fig. 2 A binary tree node implemented using 1x2 switches.

In our model, the binary address of the end-node is composed of three parts: (a) A globally unique *provider domain identifier* (PD-ID) of the provider domain of the node; (b) A *provider domain logical address* (PL-ID) which is the address of the designated POP of the end-node in the *provider domain* (PD), and (c) The MAC address of the node to uniquely identify an end-node in the access tier (DMAC as per the provider bridging standards). Each level of the logical address is used in different tiers in the network to facilitate routing, i.e. the PD-ID part is used in the core, the PL-ID is used in the provider domain, and the MAC-address is used to route in the access.

Fig. 3 illustrates the computation of a route from node 19 to node 14 in a sample binary tree using the source and destination PL-IDs. As the PL-ID indicates the state of intermediate switches while routing from the root of the binary tree to the given node, the PL-ID of node 14 is given by a string describing the state of switches *1*, *2*, *4* and *8* respectively, which is *0010*. It should be noted that the route from node 14 to the root of the tree is given by reversing the order in which the switches are encountered when traversing from the root to node 14. Note also that the state of the switch is reversed when direction of traffic flow is reversed. Therefore, the route string while traversing from a node to the root will be given by *reversing the PL-ID* and *complimenting each bit* in the reversed string.

Fig. 3(b) depicts the route calculation while traversing between node 19 and node 14 as in Fig. 3(a). First, a bitwise comparison of the source and the destination PL-ID is done. As can be seen from Fig. 3(b), the common part of the two strings (0) indicates the PL-ID of the common parent node (node 2) in the binary tree. The remaining strings indicate the states of switches from the common parent switch in the binary tree to the source and the destination along their route. To calculate the route from the source to the switch, the common bit-string is removed from the source address. The remaining string is reversed and subjected to a *1*'s compliment operation. The reversed string indicates the state of intermediate switches while traversing from the source node to the root up till the common parent. The last bit indicates the state of the common parent switch when traveling from the source to the root and this bit is again subjected to a 1's compliment operation to change the state of the switch at the common parent.

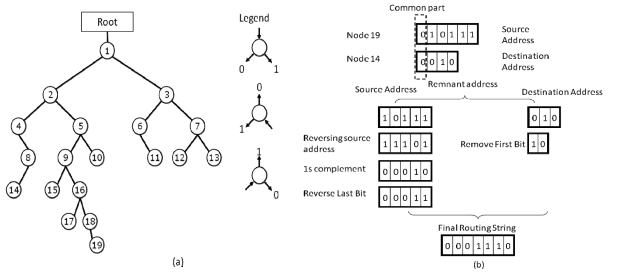

Fig. 3 (a) Sample binary tree, (b) Route calculation from node 19 to 14.

The bit-string obtained from the destination address after removal of the common part is the route from the common parent to the destination in the binary tree. The first bit in this string indicates the state of the common parent switch when routing from the root to the destination and is removed as the routing logic at the common parent is obtained from the source bit-string. The remaining string is appended to the bit string obtained from the source address after the transformation described above to get the route. In case the bit string obtained from the destination address after removal of the common part is empty then the last bit from the transformation of the source string is removed to get the route.

III. End-to-end all-Ethernet Provisioning

Having illustrated the key ideas for our concept, a number of questions arise: how does an end-node obtain its address and the addresses of other nodes; how is routing performed in the core and how is end-to-end flow managed. To address these questions, we with the overview of VLAN tags and their use as explained by the IEEE 802.1Qay standard [3]. In addition to the Customer, and Service tags as proposed by the standard, we propose to add: A-tag (address tags) and T-tag (type tag).

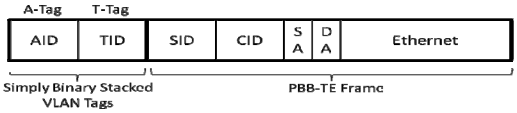

Fig. 4 Summary of new VLAN tags added to the standard Ether frame (SID: Service ID, CID: Customer ID, SA: Source Address, DA: Destination Address).

A. Ethernet Lookup Service (ELS)

Since the routing mechanisms in our model rely on logical topology information, an *Ethernet lookup service* (ELS) is required to determine the logical binary addresses corresponding to MAC addresses. Each end-node with its unique MAC address serves as a key for the lookup. A query against a MAC address in the ELS server returns the PD-ID and the PL-ID of the end-node if the node is in the same domain and only the PD-ID if the end-node is in a different PD.

If the MAC address of the destination is unknown, then the end-user generates an Ethernet frame whose payload is the ELS query. The destination address (DA) of this frame is the MAC address of the closest ELS server, which is typically the *First Hop POP* (FHP) see Fig. 5. The ELS system we propose follows a hierarchical order. Each FHP has an ELS server instance which is a child to another ELS server. Each PD has a central ELS system, and ELS systems of multiple PDs frequently exchange table updates using the IS-IS protocol. PL-ID information of end-nodes is not exchanged in this update to maintain compartmentalization. The first ELS lookup is made at the FHP which looks at its local server to determine the PL-ID of the end-node. If the FHP does not have the mapping, it queries its parent ELS server for the information. The parent can either respond to this query using its ELS database or can query a *higher* ELS server for the information. This process can continue up to the central ELS system for each PD which contains mappings for all Ethernet MAC addresses. End-node *join* or *leave* notifications are also sent to the central ELS server to maintain up-to-date MAC mappings in the system.

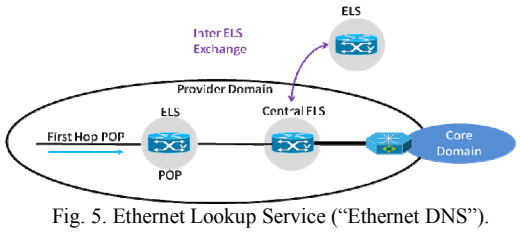

Fig. 5. Ethernet Lookup Service ("Ethernet DNS").

A control packet sent for an ELS query is differentiated from a data packet by the TYPE tag or T-tag (Fig.4). ELS synchronization and management is supported by the control plane functions, such as GMPLS, which also disseminate binary addresses for topology changes and embed routing information (address) in Ethernet frames.

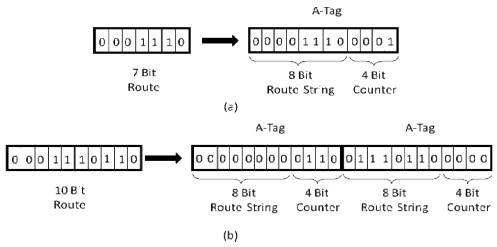

Fig. 6. A-Tag format for storing routes in VLAN tags.

B. Embedding Binary Routing

To store a route in an Ethernet frame, the A-tag is divided into an 8 bit address section and a 4 bit counter. The route string is stored in the address section while the counter is used to determine the bit in the address string to be used to do routing in the network. It should be noted that the bit-string describing the route is not a fixed length string. If the route-string is of length n with (n<8), the bit string is left-padded with (8-n) 0's and the counter variable is set to use 8-n to indicate the starting position of the actual route string. If the route string is greater than 8, the route string is broken into segments of 8 or less bits and stored in multiple *stacked* A-tags. The Canonical Format Indicator (CFI) bit is *set* thus indicating multiple A tags are required. The A-tag format is illustrated in Fig. 6. Fig 6(a) shows a 7 bit string inserted into an A-Tag. The string is left-padded with a single *0*, and the counter is set to *1* to indicate the index of the bit from which the actual route starts. Fig 6(b) depicts a *10* bit string appended into *2* A-tag's. The latter A-tag contains the last *8* bits of the route with the counter set to *0* and the first A-tag contains the first *2* bits from the original route string with six *0*'s left padded and the counter set to *6*.

If the DA is in the same PD, as was the case shown in Fig. 3, *local binary routing* is followed, while if the DA is in another

PD, *inter-provider routing* is followed. When a packet's DA is not in the same PD, it is by default sent to the root of the binary tree. The FHP computes the route to the root of the binary tree. The root of the domain then uses the destination PD-ID to generate a new bit-string to route in the core using the same principles of binary routing. The packet now reaches the root of the domain that contains the destination. This root computes the final bit-string that enables the packet to be sent to the destination FHP. In summary, end-to-end routing through provider domains involves a 3-stage process: (i) source-node POP to local border core node of the provider domains, (ii) inter-provider routing and destination address resolution, (iii) local binary routing from root of the destination provider domain to destination node. Also note that inter-provider routing typically involves traveling through multiple provider domains.

Algorithm 1: Network graph conversion

Input (Graph $G(V, E)$, root node V_R)
Output (Binary Graph)
for *each node* V_i *in* V **do**
  Determine the parent node the set of children D_i;
  while $(D_i > 2)$ **do**
    if $(D_i$ *is even*$)$ **then**
      Create $(D_i/2)$ virtual nodes
      Each virtual node is connected to 2 unique children and parent V_i
    else
      Create $(\frac{D_i}{2} + 1)$ virtual nodes
      Each virtual node is connected to 1 or 2 unique children and parent V_i
    end
    Determine the new set of children D_i;
  end
end

C. Converting an arbitrary graph to a binary tree graph

As previously mentioned, an irregular physical topology is mapped onto a regular logical topology based on binary trees, and then represented by 1x2 switches, ultimately used to perform binary routing. We now show our method to convert any arbitrary graph to a logical binary tree graph, whose every node is a binary node. The algorithm for converting any arbitrary graph into a binary tree begins by considering each node individually and converting each node into a *binary group*. 2 steps are assumed – step 1 that involves distributed pre-processing with complexity *O(N)* and step 2 that involves combining the preprocessed entities with complexity *O(N)*, together implying an $O(N^2)$ complexity. In the pre-processing, using the root node as the root of the binary graph, each node is associated with an ancestor and descendants. We define a binary group as a collection of nodes each of which have a maximum of 3 links, of which 1 link is connected an ancestor and 2 links to descendants. A node which has exactly 3 links (with one submerging and two emanating), is already a part of the binary tree and does not require preprocessing. All other nodes in excess of 3 links (degree of connectivity, D>3) are converted to a binary group, by inserting dummy nodes using the procedure described in the pseudo code. Note that the conversion for each node happens simultaneously in a distributed manner. The binary group obtained can be mapped onto the original node in terms of corresponding emanating and submerging links. This means that we do not add any extra links or nodes, but just develop virtual (dummy) nodes (and to support these virtual links), with this structure called as the binary group. Note that the obtained graph may not be a tree. To conform the graph to a tree, it should be ensured during pre-processing that a node is not a child to two distinct nodes, by pruning one of the links.

An example of converting a hub-and-spoke network into a binary graph is shown in Fig. 7. A more complex example of converting a mesh graph into a binary tree is shown in Fig. 8. Note that in Fig. 7, we choose the hub of the topology as the root.

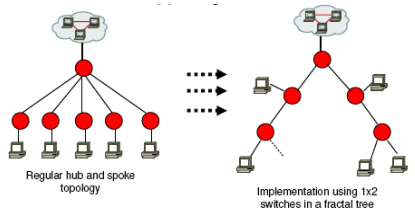

Fig. 7. Converting a hub-and-spoke topology to a binary tree.

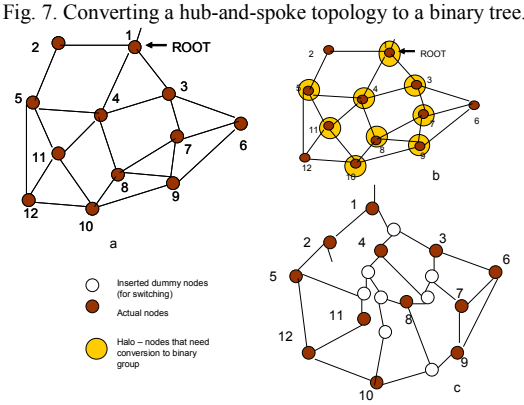

Fig. 8. Converting an arbitrary graph to a binary graph.

D. Issues of inter-domain routing: Inter-PD routing is accomplished by using a multi-path binary graph, and routing in this graph is facilitated by performing route discovery in the GMPLS control plane. GMPLS based explicit route *label switched paths* (LSPs) are assumed to be established between *one* root of every PD. Hence, an N^2 GMPLS topology is made available. Since, N is a small number (in the region of 10-30), the total number of established GMPLS LSPs are <1000. These LSPs are necessary to establish routes between PDs. We have shown how an arbitrary core network (graph) can be converted into a binary graph. Unlike a binary tree, the binary graph has multiple paths between source-destination pairs. We assume each root (in every PD) is a bridge, and using IEEE 802.1ad techniques, we disconnect all bridge ports except one (the *root bridge*), that connects the PD to the core network. This root bridge then sends Ethernet packets with S-Tags (where the CFI bit is changed to the DEI bit – drop eligible indicator). The DEI bit is set to 1, indicating that an Ethernet frame with an SVLAN sent by the root is a route dissemination frame thus well differentiated from a data frame. This frame is routed across the core using the GMPLS control plane (along each of the *N-1* LSPs from an ingress root). The frame reaches the egress root and the path is marked by that LSP number (or group of LSPs en route to the destination). Each LSP number

has the set of provider bridges (switches) that the LSP encompasses from the source to the destination. The information is decoded at the destination and converted into binary strings. A label carrying the binary route is then sent back to the source node (ingress root). The inter-PD path is hence established.

IV. Related Work

Basic binary routing has been proposed in 1970s [5]. However, structured binary routing with a logical topology has never been considered for networking. The Universal Ethernet Telecommunication Service (UETS) proposed in [6] uses a switching fabric to perform binary routing in an Ethernet domain. However, the UETS network does not use an underlying logical topology thus making routing an open issue in large scale networks [7]. The UETS network also requires a hard upgrade in all Ethernet switches inside a PD, whereas our solution with logical topologies can be implemented by a soft upgrade by using VLAN tags. UETS is not scalable as the MAC address is required to be tampered upon leading to MAC clashes. While MAC clashes can be avoided by subjecting the entire network to within a VLAN domain so that *multiple virtual router protocol* can ensure avoidance of MAC clashes, the system does not scale and poses a security threat. Further without the use of VLANs, no priority values are allowed, hence defeating the basic purpose of an-all Ethernet infrastructure.

The hierarchical addressing scheme proposed here fundamentally differs from other proposed mechanisms for hierarchical addressing such as MAC-in-MAC [8] and Q-in-Q VLANS [2], where addresses in access and metro areas contain topology information for the destination node. In provider bridged networks (Q-in-Q), the network works by forwarding packets across bridges, whereby bridges turn off STP and enable forwarding based on predefined paths. The key to those networks is the existence of predefined paths, which is non-trivial. In [9], a global Ethernet-over-WDM architecture is proposed with a combined Ethernet architecture over a modified optical node and a unified control plane; this however places restrictions on scalability. The segment based routing scheme [10] breaks a topology into different subnets and each subnet into different segments. In [11][12], a new logical hierarchy is proposed which attempts to decompose Ethernet mesh networks into hierarchical rings, and a special routing algorithm is proposed to route in the ring network. Many other past works, such as [13-17], have been focused at improving the performance of the Spanning Tree Protocol (STP) in both load balancing and recovery times. The SPB [18] algorithm has been used in PBB-TE networks assuming a flat hierarchy, which leads to scalability issues.

V. Numerical Results

To evaluate the proposed all-Ethernet solution a simulation model was built. The simulation model reflected on the aspect of logical topology design and binary routing. Traffic is generated at all nodes/ all leaves such that each frame can have every node as a destination with equal probability, and arrival process follows Poisson arrivals and exponential duration packets of MTU=1518 octets.

Our first objective is to evaluate topology design. To this end, we consider various network topologies (graphs) and convert these into a *binary* tree or graph. The topologies under consideration are based on well deployed networks – *hub and spoke*, *full-mesh*, *ring*, and *interconnected rings*. Our first performance metric is to compute the percentage of bandwidth actually utilized when we convert a physical topology to a logical binary tree. The procedure to convert a graph to a binary tree is shown in Section III. C. However, given a binary tree or graph, the question arises as to how to allocate weights to this tree so that traffic between nodes can be routed efficiently with minimal delay. To do so, we assume that a *fat-tree* is superimposed on a binary tree or graph. A fat-tree is defined as a tree where the thickness (bandwidth) of the parent link is the sum of the thickness of the child links. The bandwidth allocated in the binary tree is proportional to the thickness of the corresponding branch of the fat-tree. Shown in Fig. 9 is a comparison of the percentage of used bandwidth in terms of number of nodes while converting our sample topologies into binary trees and then applying fat-tree based bandwidth assignment. To obtain the results, we assumed load to be computed as the average number of bits per second summed over all the interfaces *at the leaves* and *the root*. Each leaf was assumed to have a 10 Mbps bidirectional link. Each generated frame was defined by its source and destination address, which were both randomly computed. The links in the fat-tree were assumed to be bi-directional. As seen in the figure, the utilization of bandwidth depends on the physical interconnection of the network. The hub-and-spoke model has very good efficiency for up to 80 nodes and then the efficiency degrades to around 50 %. This is because of the star-shaped interconnection at the physical layer, which works well for low number of nodes (in being converted to binary trees), but as the number of nodes increases, the edges towards the root are far less utilized than the edges towards the leaves resulting in lower utilization. The full mesh in contrast is the only true, monotonically increasing curve, whereby as the number of nodes increases, the utilization also increases. This is because of the well connected nature of the physical topology. A ring network has a low-bandwidth utilization, but is agnostic to the number of nodes in the network due to the fixed degree of physical connectivity of each node (=2). This limits the spreading of the binary tree and a large portion (50%) of the associated fat-tree is wasted in terms of bandwidth. Bandwidth utilization for interconnected rings is intermediate between classical ring networks and full mesh networks as expected.

Shown in Fig. 10 is the average bandwidth (and hence per port utilization) as a function of switch size (in number of switch ports). The physical N port is modeled as a virtual binary tree with N leaves and per port utilization in the graph is determined on this virtual graph. As the number of ports increases bandwidth utilization decreases, due to the self-similar nature of the traffic distribution, where local traffic does not utilize the higher order edges of the fat-tree.

Fig. 11 shows the efficiency of the system as a function of load for different implementations of the binary tree. The binary tree implemented as a fat tree with different ratios – 1, 0.8

and 0.6. A ratio of 1 implies that the sum of the bandwidths in the descendant branches (edges) is summed and then multiplied by 1 to give the bandwidth of the next branch. Efficiency is defined as the ratio of time the system is busy in transferring information normalized over the number of branches and the bandwidth of each branch. Efficiency is computed as follows. Consider a link of bandwidth 100 Mbps bandwidth implemented in out binary tree. Let the load in the network be 0.6. For the system to be 100 % efficient, the data transferred through this link would be 60 Mbps. If however, the link transfers only 40 Mbps, due to the routing mechanism then the system efficiency would be 40/60 = 0.66. In Fig. 11, as the ratio changes from 1 to 0.8 to 0.6 there is a drop in efficiency. The drop is significant as the efficiency falls from 0.8 to 0.6 implying an almost exponential drop. These results help us design our network and appropriately allocate bandwidth to links.

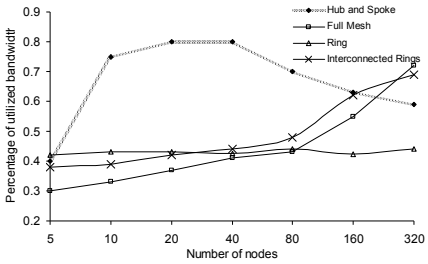

Fig. 9. Bandwidth utilization versus number of nodes.

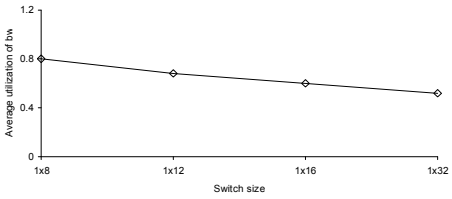

Fig. 10 Average utilization versus switch size.

Fig. 12 shows the average time required to set up inter PD paths using GMPLS and in-band control planes (bridging) as a function of average degree of connectivity. Naturally, as the degree of connectivity increases, the complexity of routing increases and the LSP computation is impacted. The convergence time is defined as the time we send a request for an inter-PD path, to the time a response is obtained by the ingress from the egress root. The computations are carried over a 25 PD network, with each PD having between 80 and 1000 nodes in its domain. The farthest distance between 2 PDs, (used for propagation delay) is 500 km, while the closest PDs are 20 km apart. For simplicity we assume bridging to follow MVRP (multiple VLAN registration protocol) to interface with the GMPLS control plane. The GMPLS control plane is out-of-band at 1.5 Mb/s bandwidth (T1). We observe that the time required for convergence is significantly lesser than an in-band control plane, especially for complex graphs (higher degree of connectivity per node) thus justifying the need for GMPLS and our chosen method for inter-PD routing.

VI. Conclusion

In this paper, we proposed a new *all-Ethernet* networking scheme in wide-area networks that involves a binary addressing and routing mechanism. The scheme is based on logical 1x2 Ethernet switches as the primary building block and it was shown that switching and routing of the unique binary addresses can be facilitated in a simple and automated fashion. The scalability is achieved by means of abstraction of the irregular physical topology onto a regular logical topology, based on the binary trees. We are able to provide end-to-end Ethernet routes using binary addresses embedded in stacked VLAN tags on native Ethernet frames. The simulation results showed the feasibility and efficiency of the proposed concept in terms of end-to-end delay and bandwidth utilization.

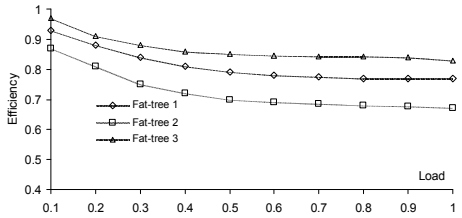

Fig. 11. Efficiency as a function of load for different fat-trees.

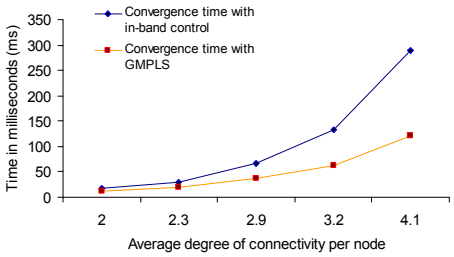

Fig. 12. Effect of control plane on inter-domain communication

VII. References

[1] Website, "Carrier Ethernet; The Technology of Choice for Access Networks," *White Paper, MEF*, www.metroethernetforum.org
[2] IEEE 802.1ad, "Virtual Bridged Local Area Networks Amendment 4: Provider Bridges," *IEEE Standard*
[3] IEEE 802.1Qay, "*Provider BackBone Bridge Traffic Engineering*" *IEEE Standard*
[4] www.tpack.com, "PBT; Carrier Grade Ethernet Transport," White Paper.
[5] A. Hopper and D. J. Wheeler, "Binary Routing Networks", *IEEE Transactions on Computers*, vol. C-28, no. 10, pp. 699-703, October 1979
[6] White Paper "Services in UETS, the 4G Infrastructure," *Ethernet Forum*,
[7] J. Barroso, G. Fernández, "UETS/EFR: A Hierarchical, Scalable and Secure Ultrahigh Speed Switching Architecture," *IEEE INFOCOM 2006.*
[8] IEEE 802.1ah, "*Provider Backbone Bridges*," *IEEE Standard*
[9] A. Hadjiantonis, et. al, "Evolution to a Converged Layer 1, 2 in a Global-Scale, Native Ethernet Over WDM-Based Optical Networking Architecture," *IEEE JSAC*, vol. 25, pp. 1048-1058, June 2007
[10] A. Mejia, et. al, "Boosting Ethernet Performance by Segment-Based Routing," *IEEE PDP '07*, pp. 55-62
[11] Y. Zhan, M. Ji and S. Yu, "Resilient Ethernet Ring for Survivable IP Metro Backbone," *IEEE ChinaCom 2006*, pp. 1-5
[12] Y. Zhan, M. Ji and S. Yu, "Hierarchical Self-Healing Rings for Mobile Ethernet Networks," *IEEE WiCom 2006*, pp. 1-4
[13] S. Sharma, et al,., "Viking: A Multi-Spanning-Tree Ethernet Architecture for Metropolitan Area and Cluster Networks," *IEEE INFOCOM 2004 Vol. 4*
[14] K. Lui, W. C. Lee, K. Nahrstedt, "STAR: A Transparent Spanning Tree Bridge Protocol with Alternate Routing," *ACM SIGCOMM*
[15] R. García, et. al, "A New Transparent Bridge Protocol for LAN Internetworking using Topologies with Active Loops," *IEEE ICPP 1998*
[16] T. L. Rodeheffer, C. A. Thekkath, and D. Anderson, "Smartbridge: A Scalable Bridge Architecture," *ASPLOS 2000.*
[17] IEEE 802.1s, "*Multiple Spanning Trees*," *IEEE Standard.*
[18] IEEE 802.1aq, "*Shortest Path Bridging*," *IEEE Standard.*