This document is published in:

IEEE International Conference on, 2013, pp. 1280-1285

DOI: 10.1109/ICCW.2013.6649434

All-path Bridging: Path Exploration as an Efficient Alternative to Path Computation in Bridging Standards

Guillermo Ibáñez, *IEEE Member*, Elisa Rojas.

***Abstract*— Link-state based routing protocols are dominant in Shortest Path Bridges (IEEE 802.1aq) and also at TRILL (IETF) Rbridges. Both standards propose a hybrid of switch and router adding a link state routing protocol in layer two that computes shortest paths between bridges. Surprisingly, path exploration mechanisms have not yet been considered at standardization bodies, in spite of some outstanding advantages: simplicity, instantaneous path adaptation to traffic load with load adaptive routing and low latency. We have developed *All-path*, a family of protocols based on simple path exploration mechanisms based on full flooding of a single frame, as an alternative to the "beaten trail" of path computation. Path exploration (either instantaneous or periodical, proactive or reactive) is an efficient alternative to path computation for bridged networks because the processing cost of address learning at bridges from broadcast frames is very low and Ethernet links provide very high link capacity so that the extra packet broadcasts do not impact load significantly. Standardization groups should consider the application of path exploration (instantaneous or periodical, proactive or reactive) mechanisms in Audio Video Bridges and in generic bridging networks like campus and data centers to find redundant paths, low latency and load distribution in simple ways instead of complex multiple path computations.**

***Index Terms*—routing, bridges, protocols, Ethernet.**

I. Introduction

SHORTEST path bridges overcome the limitations of the spanning tree protocol [1] in switched networks. Dominant approaches like Shortest Path Bridges (IEEE 802.1aq) [2] and TRILL (IETF) Rbridges [3], recently standardized, use a link state routing protocol in layer two to compute shortest paths between bridges, but these paths are shared by multiple hosts. Balancing the load at links requires complex equal cost multipath computations. It is simpler instead to find a path between every pair of hosts just-in-time, flooding the standard ARP Request frame (or other broadcast frame) through all links, snooping it at bridges with a modified address learning mechanism that associates the source address of frame to the first-arrival port and locks this association for some time, discarding duplicated packets received via other ports just by its source address.

This work was supported in part by grants from Comunidad de Madrid through Project MEDIANET-CM (S-2009/TIC-1468) .

The authors are with the Departamento de Automatica, University of Alcala, Spain (e-mail: guillermo.ibanez@uah.es).

More recently, Audio Video Bridging (AVB) Task Group of the IEEE 802.1 standard committee is elaborating amendments to the IEEE 802.1Q specification for bridges to enable time-synchronized low latency streaming services through IEEE 802.1 bridged networks. AVB poses additional challenges to bridging, such as precise timing and synchronization, stream reservation with bandwidth and latency guarantees, flow redundancy and expedited forwarding and queuing.

We found that the simple path exploration mechanisms used by All-path protocols are extremely powerful and could also help in AVB networks to find simultaneously redundant low latency paths for flow announcements. Organizations involved in bridge standardization should also explore the application of path exploration mechanisms in more generic bridging networks like campus and data centers.

To support these assertions, we describe first All-Path protocol family formed by the basic ARP-path protocol and its recent variants *Flow-path* and *Path-Moose* (aka *Tree-path*), its performance and a comparison with SPB protocols and experimental results.

II. All-path: Path Exploration vs. Path Computation

The need for bridges providing shortest paths led to the creation, by 2004/2005 of two standard groups: Shortest Path Bridges (SPB) and Routing Bridges (TRILL) aimed, among other objectives, to build switched networks of big size organized as a single IP subnet (to avoid management of IP addresses, rapidly changing in virtualized servers), while allowing full utilization of infrastructure links to obtain shortest paths.

But these standards are far from perfect, specially taking into account the variety of networks and the complexity of requirements they address. The basic routing approach in both proposals (diverging in many other aspects) is to hybridize the transparent bridges by computing paths with a shortest path routing protocol. Both SPB and TRILL use a layer two variant of the proven link-state routing protocol (IS-IS) to compute shortest path routes between bridges and to build trees rooted at bridges. This means significant complexity both in terms of computation and control message exchange and they also need additional loop control mechanisms because link state database may temporarily be not consistent (synchronized) between nodes, and additional complexity is added to obtain

path diversity by computing multiple equal cost paths for load balancing because every route between bridges is shared by many hosts. Moreover, SPB is more oriented to inter provider networks (MAC in MAC protocol) than to data centers and TRILL Rbridges are not designed to take advantage of functionalities of existing ASICs.

We have explored an alternative path to shortest path bridges, looking for simplicity and for conceptual coherency with the mechanisms of existing transparent bridges. We focused on purely bridging-based architectures for shortest path bridges, without ancillary routing protocols. We found that making use of full flooding over all links of ARP Request broadcast frames, all paths in the network are simultaneously searched in the data plane and the fastest path wins the race, assuming that a loop prevention mechanism easily discards duplicate frames arriving late to the bridges at different ports.

So, we have created and implemented *All-path*, a new family of transparent bridges (also known as FastPath and ARP-Path for the first protocol) [5][6]: a simple, low latency, zero-configuration protocol for campus, enterprise, and data center networks that uses all active links. All-path bridging protocols use a broadcast frame (the standard ARP frames or another broadcast frame) to find the path with lowest latency. An important advantage of All-path is that it automatically performs an efficient traffic distribution across redundant links. All-path, due to its simple and low latency mechanism for path set up, smoothly distributes the traffic of hosts among the redundant links in an effective way. The reason is that when a new path is set up for a host using the standard ARP, the path will be set up through the link with the lowest latency at that moment, thus avoiding the selection of the heavily loaded links.

A. *All-path Bridges vs. Standard Bridges*

All-path bridges are essentially standard bridges with a modified address learning mechanism that makes possible to broadcast frames over all infrastructure links without frame loops. Additionally, path recovery mechanisms are used to handle link or switch failures.

There are three basic differences between All-path bridges and standard backward-learning transparent bridges: First, source addresses are learnt only from ARP Request/Reply and Path Repair packets, second, but essential, the learning of source address at the port of first arrival (of broadcast frames) blocks further learning (for a short time) of the same source address at other bridge ports in order to prevent loops; third, unknown unicast frames are not replicated when the bridge has no port associated to the destination MAC ; a path recovery mechanism is used instead to rebuild an expired or broken path. The protocol requires the use of point to point links between bridges for loop avoidance but multiple hosts may share a common link to a bridge. All-path bridges can implement link aggregation, 802.1Q VLAN tagging and other IEEE standard features because the forwarding mechanism is fully independent of those features.

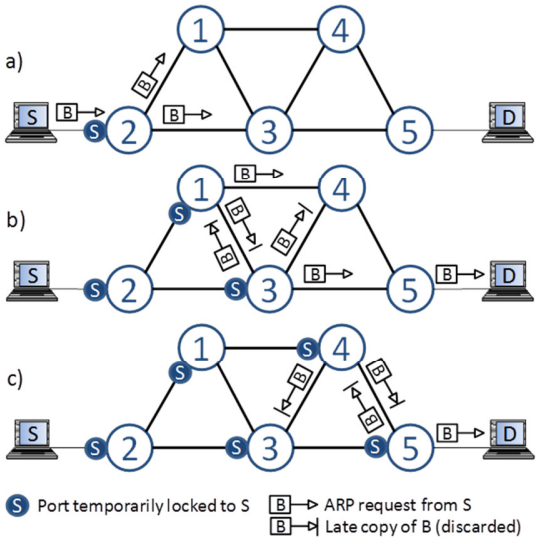
B. *BASIC ARP PATH PROTOCOL DESCRIPTION*

ARP-Path is the first protocol of a new family of transparent bridges that we identify as All-path bridges. The basic idea behind All-path bridges is simple but powerful: to explore simultaneously all network paths with a snooped broadcast frame while simultaneously preventing frame loops with the first-arrival-port to source address locked association. The first protocol variant ARP-Path [4][5], implements path set up at host level. It adapts well to campus and data center networks of small and medium size, but there are other variants possible of this new category of bridges that may adapt better to other network requirements and/or bigger network sizes.

1) *Path discovery: creation of source path*

The process, described in Fig. 1, works as follows: Source host S wants to communicate with host D and sends an ARP Request packet encapsulated into a broadcast frame to resolve the IP address of host D. The ingress bridge 2 receives the frame from S and associates the MAC address of S to the port through which it has (first) received the message, temporarily locking the learning (association) of S address to this port and preventing all other ports of bridge 2 from learning and forwarding further received broadcast frames from source address S during the lock timer interval. Thus, frames with source address S, arriving to other ports of bridge 2, will be discarded as late frames. Then, bridge 2 forwards the ARP Request frame to all ports except the one through which it was received. Bridges 1 and 3 behave as bridge 2, associating address A to the port that first receives the frame. Afterwards, bridges 1 and 3 broadcast the frame through all other ports except the port where it was first received, so that late copies of the frame arrive to 1 and 3, sent by each other. However these frames arrive at a port different from the port temporarily locked to S, so they are discarded only on the basis of its source address, only accepted at locked (associated) port.

1) ARP Request packet explores all paths, learns source address S at first-arrival port at every bridge thus creating a provisional sinking tree to S rooted at edge bridge 2.



2) The ARP Reply packet refresh at every traversed bridge the existing association (temporary lock) of S address to a port. It also sets up a confirmed association of D address to its input port.

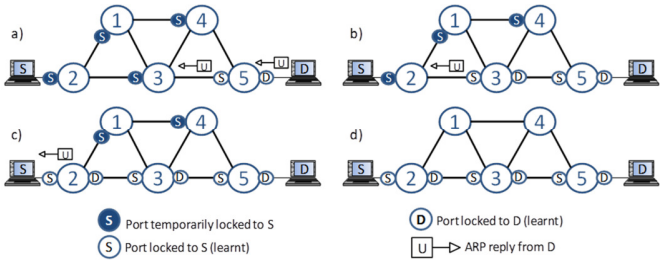


Figure 1. Path discovery: 1) S to D and 2) D to S.

The same happens at bridges 4 and 5. Hence, the temporary association (locking) of address S to a port at every bridge is propagated across the network as a tree rooted at host S, until the network edge bridges and their hosts are reached, including the host D, destination of the ARP Request. A chain of bridges with an input port locked to S is now active between S and D.

2) Path discovery: creation of destination path (ARP Reply)

The mechanism for path set up in the opposite direction is shown in Fig. 1 (2). The ARP request from S is followed by the corresponding ARP reply from D. The reply is transported in a unicast frame, sourced at D and addressed to S, and will follow back the corresponding branch of the sink tree previously set by the request. Now, ARP-Path switches take advantage of the ARP reply processing to learn the port to reach D (i.e. the receiving port of the ARP reply). As ARP Reply is transported in a unicast frame, only the switches located in the branch connecting S to D will learn about D location. The ARP Reply frame also refreshes the path from S to D.

3) Path Recovery

Established paths may get broken at some point either by the expiration of an address timer or by failure or initialization of a link or bridge. When a link connecting two bridges fails, all MAC addresses learnt at both ports ending the link are flushed. The same happens at all ports of a node, when the node reinitializes. Then, whenever a bridge receives a frame with an unknown unicast destination address (i.e. the address is not associated to any port of bridge), the path may be rebuilt from the source bridge or from the current bridge. Many variants of these two approaches can be designed; we explain here two basic methods.

In the first approach, the bridge that receives the unicast frame with unknown destination encapsulates it inside a *Path_Fail* message and returns it in the backward direction towards the source host. This message is processed at each bridge in the backward path, which forwards it via the port associated to the source host till it reaches the source edge bridge. The *Path_Fail* message is addressed to the *All_ARP path_Bridges* MAC multicast group and delivers the unicast frame looped back as payload. Every bridge in the path checks if it is the source edge bridge of the source host of the looped back unicast frame (i.e. if the host is directly connected to it).

In this case the bridge broadcasts a new *ARP Request* on behalf of the unicast frame's source host and the path is recreated in the normal way.

In the first approach, the path is rebuilt from the affected bridge onwards by issuing either a standard ARP Request on behalf of the source host or a *Path_Request* message addressed to the *All_Fastpath_Bridges* multicast address. In the former case the ARP Request is replied by the destination host with an ARP Reply that selects the path towards the failed bridge, which intercepts the ARP Reply. In the latter case, a *Path_Request* message containing the source and destination MACs and IP addresses is broadcasted in the forward direction and processed and forwarded by all the bridges traversed till the bridge attached to the destination host. D.

C. Coexistence with standard bridges IEEE 802.1 and 802.1Q

All Path switches may operate connected to standard bridges in core-island mode. A core of All Path bridges may interconnect islands of standard bridges running the spanning tree protocol. Self-configuration of islands of standard bridges operates as follows: All Path bridges connected to standard bridges receive standard Rapid Spanning Tree BPDUs on the ports connecting to the standard bridge islands. As a consequence they run the standard RSTP protocol on those ports, emitting BPDUs that announce the All-Path bridge as having a direct connection to a virtual root bridge with maximum bridge priority as described in [5]. Hence, All Path bridges are automatically selected as root bridges by the standard bridges and a number of separate trees are built rooted at the fast path core. Note that no frame encapsulation is needed to traverse the core. Rapid Spanning Tree Protocol may also be used as a fall-back or ancillary (see Path-Moose variant below) protocol.

D. VLANs

Opposite to SPBV, ARP-path has the advantage of being fully independent of VLANs. ARP Path protocol includes a robust broadcast loop prevention mechanism, so it does not need to assign VLANs to separate forwarding domains per bridge to prevent broadcast loops. VLANs work exactly like in standard switches, each VLAN behaves as independent forwarding domain. With ARP Path, VLANs can be used for any purpose in order to create fully independent virtual topologies running ARP Path coexisting with standard protocols in separate broadcast domains. As an example, a network of bridges capable of running ARP Path and 802.1D/.1Q protocols could operate at several independent VLANs as an ARP Path bridged network and as an 802.1D RSTP/MSTP network in other VLANs. This is a form of protocol coexistence over a common infrastructure without interactions (ships-in-the-night).

III. PROTOCOL EVOLUTION AND VARIANTS

The first protocol variant (described above), ARP-Path

[4][5], implements path set up at host level. It adapts well to campus and data center networks of small and medium size, but there are other variants possible belonging to this new category of bridges that may adapt better to other network requirements and/or bigger network sizes. The second variant, Flow-path [6], associates both source and destination MAC addresses to every port, thus associating a flow to a port, instead of just a source address. This variant has the advantage of guaranteed path congruency in all situations and fine grained load distribution in heavy loaded servers.

The third variant is Path-Moose [7], a simple and scalable variant that combines the path discovery of ARP Path (at bridge level) with hierarchical bridge addressing of the form *bridgeID:hostID*. Host addresses do not require modification, a private host hierarchical address is assigned by edge bridges, which perform NAT of MAC address to frame received from directly attached hosts converting universal MAC address into hierarchical address. Path-Moose bridges learn from ARP Requests only the bridgeID address prefix at ports instead of the full host address. In this way, Path-Moose builds a set of rooted sink trees, one tree rooted at every edge bridge. These sink trees serve as unicast destination paths for all hosts connected to the bridge root of the tree. Alternatively, *Set_Tree* packets can be periodically sent (e.g. every five min.) from every source bridge, learning at every bridge only the bridge identifier (part of the hierarchical MAC) instead of the full MAC address. Path recovery simply consists of triggering an immediate refresh of all trees by sending a single *Refresh_Trees* broadcast frame that is fully flooded and triggers at every bridge its tree refresh. Forwarding table size requires only an entry per bridge plus one per directly attached host. Destination paths are shared among the hosts connected to the same edge bridge, so that path recovery time is often null because once the path is recovered to reach a host, all hosts connected to same bridge will use it because only the bridge address part of the MAC address is learnt at bridges and used to reach destination edge bridge.

Finally, a MAC-in-MAC All-Path protocol variant is also possible. The mechanism of locking the source address learning to first-arrival port is used in this variant between bridges (on full backbone B-MAC addresses) as in SPBM [2]. Backbone bridges may set up trees instantaneously by sending a *B_Set-Tree* multicast frame. Backbone bridges confirm with an ack frame a new tree link to root bridge at every hop to ensure path symmetry at the backbone. Although final tree topology is not predictable (is based on latency only), the network has maximum resiliency; even if there is only a path or branch available, it will be selected. Networks where simplicity and resiliency is more important than predictability could benefit from this approach.

IV. Comparison with Shortest Path Bridges

We compare ARP Path protocol with SPB. SPBM (SPB MAC in MAC encapsulation) is oriented to Carrier Ethernet whilst ARP Path and SPBV (SPB Q-in-Q) are oriented to campus and data centers. We consider a network of b nodes (bridges), E edges (links), and h active hosts and assuming that $h >> b$ (h between one and two orders of magnitude bigger than b).

Table I shows a summary of this comparison that is illustrated below.

TABLE I. All-Path vs. Link-State Comparison

	Link State (SPB)	All Path
Forwarding state (CAM)	*O(b+h)*	*O(h)*
Routing state	*O(b*d+h)*	*≤ O(h)*
Number of messages	*O(b*E)*	Standard ARP messages + extra flood: *h*(E-N+1)*
Computational complexity	*O(b*log(b)+h)*	One CAM look-up (MAC, port)
Convergence time	*O(path length bs)*	*Negligible (extra processing of ARP at ARP Path bridges)*
Fault recovery	Messages O(2*E) Time *O(path length bs)* Recompute *O(b*log(b))*	Messages *O(2*E*hostlink)* Time *O(path length bs)* Recompute *O(b)*
Path diversity	*O(b*b*log(b))*	*O(b)*

A. *Forwarding and forwarding state.*

At every bridge, the link state protocol needs a routing table input per bridge or host. That is *O(b+h)*, equivalent to *O(h)*. ARP-Path uses an input per active host but only at bridges of the active paths, which are a fraction of all bridges (s/b, being s the average path length in hop count minus one). Regarding routing state the information needed to compute routes is *O(b*d+h)* where d is the average node degree.

Computational complexity

Shortest Path Bridging (SPB) uses the link state protocol IS-IS, to acquire the network topology and then apply the Dijkstra shortest path algorithm to compute shortest path routes. Its computational complexity is, for a network of b bridges, $\theta(b^2)$ with a minimum of $b \bullet log(b)$. The ISIS SPB protocol used in Shortest Path Bridges is even more complex than IS-IS. The reason is that paths between every pair of nodes must be congruent (must coincide in both directions). If a path is not congruent, the backward learning mechanism does not work properly and paths may oscillate (i.e. flap). To prevent this and provide multiple paths between bridges, IS-IS SPB computes all Shortest Path Trees of all nodes at every node [8]. The computational complexity of the Dijkstra algorithm is then multiplied by b resulting in $b2 \bullet log(b)$, that may affect scalability and reconfiguration times in large networks. IS-IS SPB implements multipath routing between bridges to distribute load per flows via parallel paths between bridges assigned to different flows. Instead, All-path sets up on demand low latency paths between hosts when needed so that the flows obtain diversified paths without additional complexity.

B. *Convergence time and latency*

In SPB the convergence time is the time needed to receive all messages required to achieve convergence which is proportional to maximum path length in the topology. The nodes need to synchronize data bases. The paths are precomputed, no added latency to path. In ARP Path, convergence time is per flow and depends on the shortest path

between nodes. Path latency is not increased versus ARP Request as it is "snooped" at the bridge in parallel with ARP process, selected path is the one with lowest latency in forward direction.

C. *Fault recovery. Number of messages*

In case of link failure, in SPB the two adjacent nodes redistribute the new link state to all bridges. In All-path Path-requests are broadcasted to all bridges. Values are shown in table. ARP Path bridges do not periodically exchange routing information. Instead, the standard ARP Request and Reply message exchange is reused to set up paths when needed. All-path bridges do not have additional message overhead, apart from the extra ARP Request traffic described below. SPB distributes local information plus host list to other bridges. Values are as shown in table: *O(b*E)* if all information is sent in one message by every bridge. ARP Path sends and ARP Request to all neighbors: *O(h*E)*, but these message are reused, SPB also uses them, marginal cost is zero. ARP Path slightly increases the number of broadcast frames compared to spanning tree. The increment is numerically low because the majority of links are normally direct and single access links to hosts; hence, they are not redundant and will forward broadcast frames in both cases, with spanning tree and with All-path. All-path bridges broadcast ARP Request frames over all inter switch links, instead of only via spanning tree links. The increment in frames is twice the number of redundant inter switch links, a relatively small value. For the data center network of fig. 5 with 250 hosts (i.e. 250 host links), 10 distribution switches, 4 core switches and 26 inter switch links, the total number of broadcast frames would be 263 with spanning tree and 289 with All-path.

V. SIMULATIONS

A. *Load distribution. Regular mesh topology*

All path protocol has been implemented in OMNeT++ simulator. Figure 2 shows the average results of two runs (10000 seconds each run) in a 3x3 grid network (100 Mbps link rate) and flow inter-arrival time (1/λ) of 1.6 seconds. To show the protocol load distribution capabilities, the percentage of flows originated at node 0 and destined to node 8 (edge to edge) and the total link load utilization (in each direction) is provided for each link. Figure 4a shows how the flows from node 0 to node 8 are evenly distributed (by halves at every node) among the four main paths spanning from bridge 0 to bridge 8 when no other traffic is exchanged in the network. Fig. 2b shows the loads at links with background traffic in two different scenarios: when a uniform distribution of traffic between all source and destinations is used (left) and with traffic biased towards node 6 by a factor of 4 (node 6 has four times more probability to be chosen as source or destination than any other node). Fig. 2c shows percentage of 0-8 flows at every link with load balance, although not as precise as with the flow model. The mesh on the right shows how the protocol adapts to the traffic conditions diverting some flows from 0 to 8 to paths away from node 6. Load distribution has also been verified at other topologies with equal cost paths like the one shown in next section.

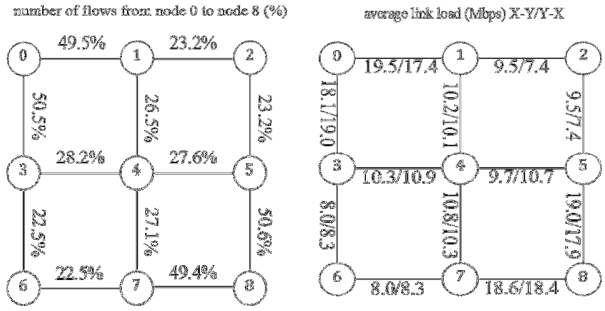


Fig. 2 a. Load and flows distribution (nodes 0 to 8) without background traffic.

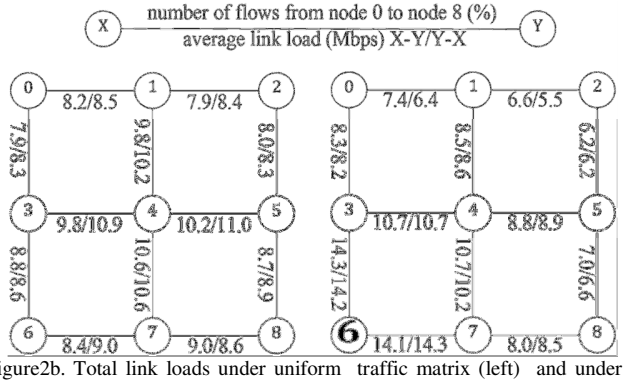


Figure2b. Total link loads under uniform traffic matrix (left) and under a traffic matrix biased towards node 6 by a gravity factor of 4 (right).

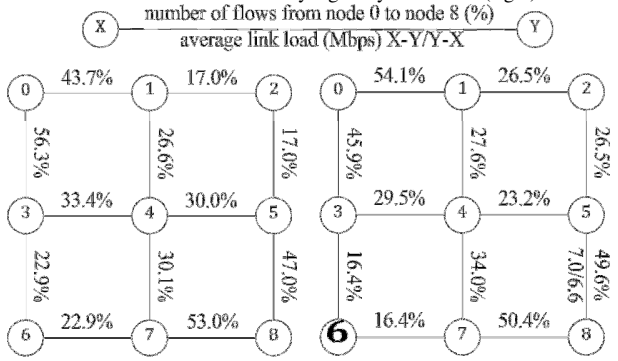


Figure 2c. Distribution of the flows between node 0 and 8 with uniform traffic matrix and under a traffic matrix biased towards node 6 by a gravity factor of 4 (right)

B. *Latency*

Latency was evaluated through simulation on the 250 host network of Fig. 3 (25 hosts connected to every HS switch) with a flow generator during 5000 seconds. Traffic is randomly distributed among all servers with equal probability (packet sizes have Pareto distribution). All links have 100 Mb/s speeds. The switches are modeled with 2 us processing time. We compare SPB and All path protocols in same scenario. Latencies are shown in Fig. 4 in seconds in logarithmic scale.

SPB (without multipath computations) exhibits latencies more than one order of magnitude higher than All-path and up to one order of magnitude average maximum latencies at servers due to the use of alternative paths. Minimum latencies

are mostly determined by processing time at switches and propagation delays. Extensive simulations are currently performed to evaluate latencies at high loads.

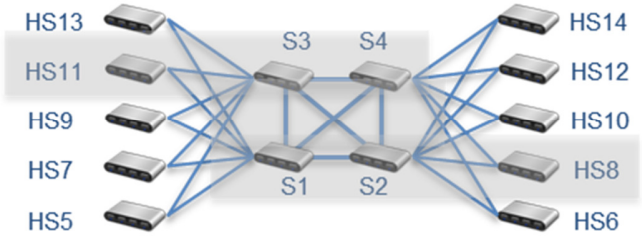
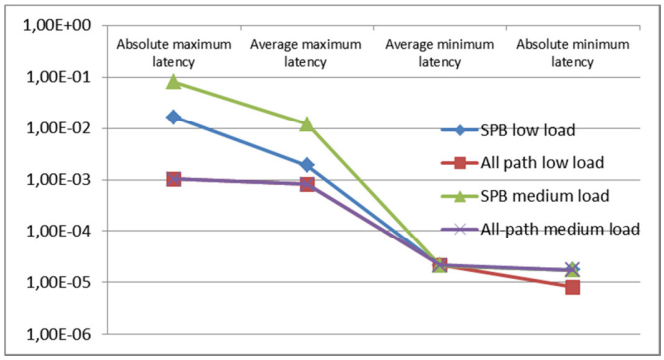


Figure 3. Topology for latency comparison SPB vs. ARP-path .



VI. FIGURE 4. PACKET LATENCIES OF DATACENTER NETWORK WITH 250 SERVERS.EXPERIMENTAL RESULTS

Any protocol enabling all redundant links at layer two must verify its robustness against path fails and broadcast loops, that can produce network meltdown. ARP Path has been succesfully implemented in a variety of platforms: Linux using *ebtables* and Openflow [9] and validated in real world scenarios with hosts connected to Internet via university campus networks. The simplicity of the protocol facilitated the implementations. After validation in previous platforms, All-path protocol was implemented on NetFPGA[10]. The internal latencies obtained are those typical of a switch implemented on a NetFPGA. The effect of load distribution has been verified in a single square four-node network (Fig. 5). Flows from hosts connected at one node to hosts attached to the opposite node were established with *iperf* and it was demonstrated that load can reach the maximum link limits with All Path, whilst with spanning tree protocol one link is disabled to prevent loops, cutting one of the two parallel paths and thus limiting the maximum per flow capacity to half (500 Mbps).

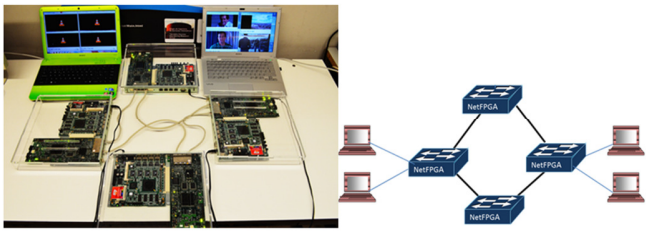


Figure 5. Four All-path switches network on NetFPGAs

VII. RELATED WORK

Besides the above mentioned standard protocols TRILL [3] and SPB [2] that use specific variants of IS-IS routing protocol, SEATTLE [11] also uses link state routing together with a DHT-based directory for host resolution to replace ARP and suppress broadcasts. Broadcast reduction is a generic issue of switched networks that is discussed at IETF. A generic mechanism to reduce broadcasts is based on implementing ARP proxy function at edge bridges (top of rack switches) as explained in [12].

An approach that is conceptually close to our approach is described in [13].

VIII. CONCLUSION AND FUTURE WORK

All-Path bridges evaluations and implementations show the performances that can be obtained with simple path exploration mechanisms based on layer two flooding. All-path protocol family shows different variants that adapt to different network sizes and requirements. Due to its extreme simplicity and high performance, its application to AVB and in generic switched networks should be investigated by the standardization groups involved. Simulation results show that they provide very efficient link utilization and load distribution among alternative paths without additional mechanisms. These properties make it valuable as a simple, low latency, high throughput mechanisms for enterprise, data centers and AVB networks.

REFERENCES

[1] IEEE 802.1D-2004 IEEE standard for local and metropolitan area networks-Media access control (MAC) Bridges.

[2] Shortest Path Bridging. http://www.ieee802.org/1/pages/ 802.1aq.html

[3] Transparent interconnection of lots of links (TRILL) WG. http://www.ietf.org/html.charters/trill- charter.html.

[4] G. Ibanez et al. “FastPath Ethernet Switching: On-demand Efficient Transparent Bridges for Campus Networks“. LANMAN May 2010.

[5] G. Ibanez et al. ARP Path: ARP-based shortest path bridges. IEEE Communication Letters. July 2011

[6] E. Rojas, G. Ibáñez, D. Rivera, J. Carral.” Flow-Path: An All-path Flow-based Protocol”. Proceedings of LCN 2012. pp. 244-247.

[7] G. Ibáñez et al. “Path-Moose: A Scalable All-Path Bridging Protocol. IEICE Transactions on Communications: Vol.E96-B, No.03, Mar. 2013.

[8] J.Farkas, Z. Arató. Performance analysis of shortest path bridging control protocols. GLOBECOM 2009 Honolulu pp. 4191-4196.

[9] G. Ibáñez et al. A Simple, Zero Configuration, Low Latency Bridging Protocol. LCN 2010 demos. http://www.ieeelcn.org/prior/LCN35/lcn35 demos/lcn-demo2010_ibanez.pdf

[10] E. Rojas et al. “Implementing ARP-path low latency bridges in NetFPGA”. Proceedings of the ACM SIGCOMM 2011 conference pp. 444-445. ACM New York.August 2011. doi>10.1145/2018436.2018512

[11] C. Kim, M.Caesar, J. Rexford. Floodless in SEATTLE: A Scalable Ethernet Architecture for Large Enterprises. ACM SIGCOMM Computer Communication Review vol. 38 no.4, pp3-14. Oct. 2008.

[12] A. Ghanwani, H. Shah, N. Bitar.” ARP Broadcast Reduction for Large Data Centers. https://datatracker.ietf.org/doc/draft-shah-armd-arp-reduction/

[13] K. Miyazaki.; K. Nishimura.;J. Tanaka.;S. Kotabe;"First-Come First-Served Routing for the Data Center Network: Low Latency Loop-Free Routing," World Telecommunications Congress (WTC), 2012 , vol., no., pp.1-6, 5-6 March 2012