Álvarez Horcajo, J., Ibanez, G., Constantin, B.N., Martínez Yelmo, I. & Arco, J.M. 2019, "Scaling and Interoperability of All-Path with Bridged and SDN Domains using VXLANs", in 2019 IEEE 44th Conference on Local Computer Networks (LCN), Oct. 14-17 2019, Osnabrück, Germany, pp. 97-100.

Available at http://dx.doi.org/10.1109/LCN44214.2019.8990814

Scaling and Interoperability of All-Path with Bridged and SDN Domains using VXLANs

Joaquin Alvarez-Horcajo, Guillermo Ibanez, Boby N. Constantin, Isaias Martinez-Yelmo, Jose M. Arco
Departamento de Automatica, University of Alcala, Alcala de Henares (Madrid), Spain.
Email: j.alvarez@uah.es, guillermo.ibanez@uah.es, boby.constantin@uah.es, isaias.martinezy@uah.es, josem.arco@uah.es

***Abstract*—All-Path protocols, namely ARP-Path and TCP-Path, provide shortest path bridging by using path discovery and backwards learning in meshed topologies. However, their domain size may be limited to prevent excessive Layer 2 (L2) broadcast traffic overload in hosts. This paper proposes the use of Virtual Extensible Local Area Network (VXLAN) to solve this issue. Moreover, this paper also verifies the extensibility of All-Path domains and its interoperability with other different L2 protocols via VXLAN, which enables flexible network hybridization. Although encapsulation via VXLAN is heavier than other standard protocols designed for L2 scalability, the overall advantages and suitability for virtualized networks are excellent. Results show full compatibility and interoperability combined with good throughput and delay performance.**

***Index Terms*—All-Path, VXLAN, STP/RSTP, SPB, TRILL**

I. Introduction

The limitations of L2 domains in networks are low utilization of the infrastructure and lack of scalability. The basic bridging mechanism of frame diffusion and MAC address learning is heavily restricted by the Spanning Tree Protocol/Rapid Spanning Tree Protocol (STP/RSTP). It disables all redundant links in the infrastructure, prevents shortest path switching and forces unnecessary congestion in the spanning tree. Moreover, the small range of Virtual LAN (VLAN) makes impossible the adequate separation of multiple tenants traffics in data center networks. Alternative standard protocols began in 2004 at IEEE and IETF to solve these limitations by allowing larger L2 domains belonging to a single IP subnet. These protocols were Shortest Path Bridging (SPB) [1], and TRansparent Interconnection of Lots of Links (TRILL) [2]. However, during their development, VXLANs were implemented and documented as RFC7348 [3]. VXLAN addresses the need for overlay networks within service providers and enterprise data centers to accommodate multiple tenants (providing isolation) and scale-out their L2 domains. While the approach of SPB and TRILL was to create large L2 switching domains by adding encapsulation and routing in L2 (MAC-in-MAC), the VXLAN approach was to add encapsulation in layer 4 (UDP) and perform routing in Layer 3 (L3) using IP multicast. This approach creates a virtual L2 domain that can physically reside in different L2 subdomains located at even different data centers.

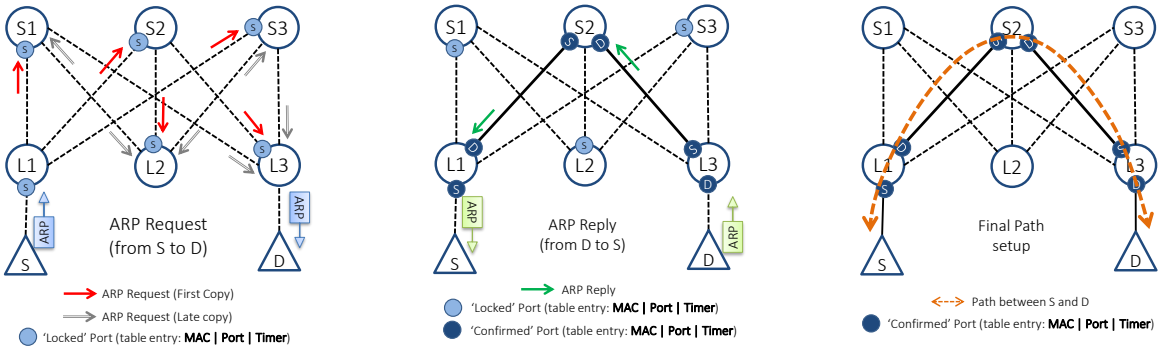
This paper describes how VXLANs enhance the scalability of L2 All-Path domains [4], [5] and achieve full interoperability with other domains i.e. legacy or SDN domains. Section II describes ARP-Path (first All-Path protocol), VXLANs and their integration, Section III shows a proof-of-concept of the proposal and its validation. Section IV resumes the related work and, section V contains the conclusions from this work.

II. All-Path Integration with VXLANs

All-Path protocols [4], [5] are implemented in a hybrid All-Path/SDN software switch (AOSS) derived from the OpenFlow software switch BOFUSS [6]. ARP-Path can also be implemented in a variety of hardware (NetFPGAs, P4 targets, etc.). Hence, it is possible their use in modern hardware devices. We explain in the next section, for simplicity, the behaviour of ARP-Path, the first developed All-Path Protocol.

A. ARP-Path operation

ARP-Path is an evolution of the Ethernet transparent bridges. It provides a simple, shortest path bridging, which is transparent to hosts and does not require a spanning tree or a link-state protocol for routing. ARP-Path sets up paths between hosts by reusing standard Address Resolution Protocol (ARP) frames. Its behaviour in a L2 domain is illustrated in Fig. 1. First, it is necessary the flooding of an ARP Request packet. In Fig. 1a, switch L1 sends the broadcast frame through all links. The rest of switches receiving the broadcast frame establish a lock according to the source MAC address (S) to the input port and forwards it through all links afterwards. Frame loops are prevented by the modified learning mechanism that locks the relearning of the source address (S) at other ports for a locking time and allows to discard broadcast frames from the same source MAC address (S) received via a different port than the locked one. The locked ports (light circles) on each switch triggered by the ARP Request frame are labelled with the source MAC address (S). Later, once the ARP Request reaches the destination host through the fastest path, it replies with a unicast ARP Reply frame (Fig. 1b). This frame is forwarded back to the source host using the previously learned fastest path by using the locked ports to reach the source. Then, it is learned the D MAC address and the S locked ports change to learned in the fastest path, which completes the address learning for both directions. Learned ports are in dark circles labelled with its learned MAC address in Fig. 1b. The rest of the locked ports lose their lock after expiring the locking time. This procedure allows normal unicast communication as shown in Fig. 1c.



(a) Network exploration (b) Fastest path confirmation (c) Established Path

Fig. 1: ARP-Path operation example

B. VXLANs standard

VXLANs addresses three main issues, as stated in RFC 7348 problem definition: to implement isolation among network users (multitenant data center) to allow the independent assignment of MAC addresses and VLANs ID without duplication; to use IP for interconnection (e.g. to obtain efficient and scalable multipath routing via Equal Cost Multipath Routing (ECMP)), avoiding disabled links by using a L3 overlay.

C. ARP-Path operation on VXLAN

It is worth noting that ARP-Path protocol uses the standard ARP Request and ARP Reply messages to find reactively the shortest path, no protocol-specific messages are used for path set up (but protocol-specific messages exist for path repair). In Fig. 2, if a host in an All-Path Domain wants to establish a communication with a host in other domain, the host sends out an ARP Request frame for the other host. When this broadcast frame reaches the VXLAN Tunnel End Point (VTEP), this VTEP has not yet a mapping for the requested host and encapsulates the ARP request in an IP multicast packet and forwards it as a VXLAN multicast group datagram, which is sent to all VTEPs belonging to the same VXLAN. The encapsulated datagram has the IP address of the source VTEP as the source IP address and the VXLAN multicast group as the destination IP address and the packet is distributed to all VTEPs of the multicast tree. All VTEPs decapsulate the packet and check if it is a valid VXLAN frame, in such case, they forward the ARP Request to their local network, they also learn the IP address of the source VTEP from the outer IP address header and inspect the packet to learn the MAC address of source host placing this mapping in the MAC-addresses local table. When the destination host receives the ARP Request answers with an ARP Reply with its own MAC address and learns the MAC and IP from the source host. When its VTEP receives the ARP Reply, it already knows the source VTEP from the previously received ARP Request frame; thus, it can use a unicast VXLAN tunnel to forward the ARP Reply back to the source VTEP. When the source VTEP receives back the encapsulated ARP Reply frame, it de-encapsulates and forwards the ARP Reply to the source host and also learns the IP address of the destination VTEP from the outer IP address header and inspects the original packet to learn the MAC and IP addresses from the destination host. Subsequent IP packets between the source and destination host are unicast forwarded, based on the mapping information on the source and destination VTEPs. VTEPs can also behave as proxy ARPs to reduce the flooding over the transport network.

D. Link Failure Recovery in VXLAN scenarios

VTEPs in VXLANs scenarios, due to its gateway function, are always located in key positions such as hypervisors, ToR switches or gateways. In all cases, they indirectly participate in the recovery of established flows between different domains after a link failure. If each domain repairs locally the broken paths by link failures, the established flows will be unaffected.

III. Validation

The objectives of the validation are two-fold. First of all, to demonstrate how the exploration mechanism of All-Path protocols is feasible in different connected domains using VXLAN encapsulation. Secondly, to verify the interoperability among L2 domains running different L2 forwarding protocols, such as *TCP-Path*, *ARP-Path*, SDN or standard STP domains.

A. Testbed and Experimental Setup

The hardware infrastructure consists of five computers powered by Intel(R) Core(TM) i7 processors with 24 GB of RAM, which are all interconnected via an IP network. Furthermore, we use the well-known *Mininet* framework as our emulation tool to define the topologies for our experiments.

Our VXLAN scenario is composed of five different L2 domains with 100 Mbps links, which are shown in Fig. 2. The ”All-Path D” and ”All-Path C” domains use the software switch with both support of ARP-Path and TCP-Path [4], [5]. The ”All-Path D” domain uses the distributed recovery mechanism whereas the ”All-Path C” domain uses the centralized recovery mechanism, both proposed previously in [5]. The ”ONOS domain” is an SDN domain with an ONOS controller [7] that uses the reactive forwarding application based on OpenFlow to manage a topology of Open vSwitch (OvS) devices. The ”ECMP domain” is also an SDN domain

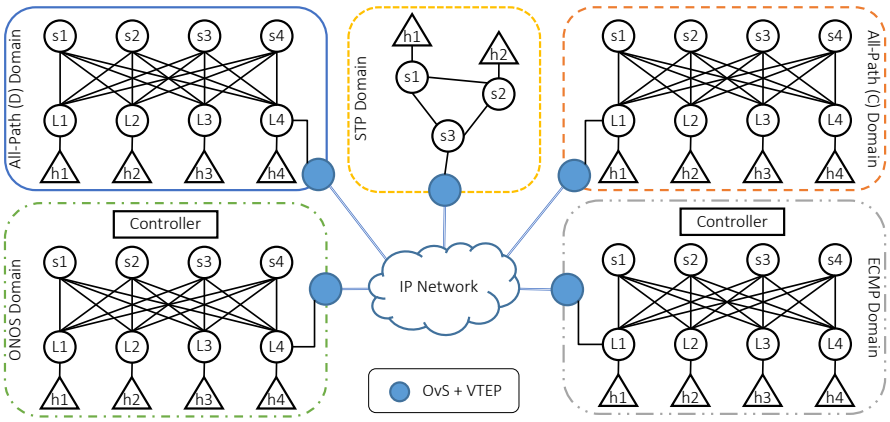


Fig. 2: VXLAN interconnecting different L2 domains

TABLE I: Inter-domain binary rate (Mbps)

Mbps	**All-Path D**	**All-Path C**	**RYU**	**ONOS**	**STP**
All-Path D	95.7	95.1	95.0	95.1	93.6
All-Path C	95.1	95.7	93.1	93.1	93.1
RYU	95.0	95.3	95.6	95.3	93.1
ONOS	95.1	95.1	95.3	95.4	93.6
STP	93.6	93.6	93.1	93.6	94.2

TABLE II: Inter-domain ARP Round Trip Time (ms)

ms	**All-Path D**	**All-Path C**	**RYU**	**ONOS**	**STP**
All-Path D	0.66	1.28	1.53	3.50	1.00
All-Path C	1.17	0.59	1.56	3.57	1.05
RYU	1.65	1.64	0.90	4.10	1.53
ONOS	3.58	3.64	3.94	2.16	3.69
STP	0.97	1.00	1.37	3.32	0.68

with OvS devices managed by a RYU controller instead of an ONOS controller. The RYU controller uses the OpenFlow protocol to establish paths by using an ECMP policy. Finally, the fifth domain uses three real switches running STP to avoid loops. The first four former domains use a 4x4 *Spine-Leaf* network topology [8], and the last domain uses a triangular topology. Each domain is connected through a VTEP node (blue circles in Fig. 2), which is implemented with OvS.

B. Results

Firstly, to verify the interoperability among the different L2 subdomains, different flows are sequentially established between all the domains, including flows between hosts from the same domain. Table I summarizes the binary rates obtained from using the *iperf* tool in the communication between the different hosts. The binary rates are closed to the 100 Mbps links. Thus, the hosts from any domain are able to connect with the hosts from the other domains without losing significant performance. Finally, the bit rates from inter-domain flows are very similar but slightly smaller than the bit rates of the intra-domain flows since the intra-domains flows have smaller queuing, propagation and transmission times.

Secondly, we measure the establishment time between hosts located in different domains. Table II shows the Round Trip Times (RTTs) from ARP processes (ARP Request + ARP Reply) between hosts from the different domains by using the *arping* tool. We can observe how the RTTs from the ARP procedures are quite different. Observing first the intra-domain ARP RTTs, we can see how the lowest ARP RTT measured corresponds to the STP domain because it is implemented with hardware switches and it has a smaller size than other domains. Results for All-Path domains are close to those from the STP domain if we consider their bigger size. Additionally, we can see how the ONOS domain introduces the biggest delay because the ARP proxy functionality is not enabled by default on the controller, reducing the efficiency of the network when processing ARP traffic. This effect is not observed in the ECMP domain, which does not have either any ARP proxy functionality, but it manages the ARP traffic in a more efficient way rather than the ONOS controller. Moreover, we can observe how the inter-domain ARP RTTs are always higher rather than the intra-domain ARP RTT of the source and destination domains as expected. Indeed, they are approximately the addition of the RTT in each domain. Obviously, they cannot be the same since there are many involving factors (processing time, queuing time, etc.) but their proximity demonstrates the achieved connectivity between the different domains through the use of VXLAN. Particularly, the largest delays always involve the ONOS domain since it requires more time to forward the ARP packets.

Finally, we evaluate the performance of the distributed and centralized path recovery mechanisms supported by the All-Path protocols in combination with extended L2 domains via VXLAN. ”All-Path D” domain uses the distributed recovery mechanism and ”All-Path C” domain uses the cooperative recovery mechanism from [5] as we stated before. The evaluation is performed as follows: a flow is established between hosts from different domains using a uniform distribution with an Inter-Arrival Time (IAT) of 0.104 ms. After a random time, one link of the flow path is turned off in the destination domain to force the execution of the recovery mechanism in that domain. By measuring the dropped packets until the recovery, the recovery time is obtained. The results in Table III show how both recovery mechanisms work as expected over a VXLAN extended local network but the distributed recovery mechanism achieves smaller recovery times than the cooperative recovery mechanism because of the small load of the network. This fact

TABLE III: Recovery Time (ms)

ms	**All-Path D**	**All-Path C**	**RYU**	**ONOS**
All-Path D	—	4.35	82.26	65.01
All-Path C	2.19	—	82.27	64.03
RYU	1.95	4.25	—	111.21
ONOS	2.79	5.48	80.83	—
STP	1.55	4.72	20.37	45.75

is due to the load of the network is small. Thus, the queuing, transmission and processing time of the distributed recovery is smaller than the processing time needed by the controller in the cooperative recovery [5]. Moreover, we can see how path recoveries based on SDN solutions are slower since they follow reactive solutions without pre-establishing alternative restoration paths. Hence, they need to compute an alternative after a failure. Then, they remove the old rules and install the new ones according to the new updated path. Results for failures in the STP are not shown since they are one order of magnitude bigger since RSPT is not supported by the hardware switches.

IV. Related Work

There are two basic approaches for scalability: create an overlay routing on L2 or L3. We can say that SPB and TRILL use encapsulation and routing on L2 and focus on scaling up the size of the L2 domain, while VXLANs and other similar L3 overlay solutions scale out the number of L2 domains by multiplying the number of interconnected L2 subdomains, which all together make a full unique L2 domain. The L2 protocols considered in this section have two main objectives: to increase the scalability of the L2 domains and to solve the limitations imposed by the spanning tree protocol such as blocking of all redundant links and its consequences (nom shortest paths, etc.). We limit our analysis to revise the scalability problem in the approaches that have reached either significant adoption or standard status.

Two competing protocols have reached the status of network standard: SPB [1] and TRILL [2]. Both protocols scale up the L2 domains employing an additional L2 encapsulation. Both implement a large L2 domain belonging to a single IP subnet, but it is not possible to move parts (e.g. a virtual machine) of the L2 domain across the IP network because the problem definition at the time of development did not consider it necessary. Both use an adapted IS-IS protocol for routing in L2. SPB focuses on large backbones domains and TRILL focuses on self-configuration in enterprise data centers.

Overlays using L3 (IP) for routing belong to a different family of protocols than L2 overlays, although they share the objectives of domain extension and scalability. They connect multiple L2 domains with overlays and tunnels to increase scalability. VXLANs achieve scalability by scaling out the number of L2 domains, not their size. Layer 3 Virtual Private Network (L3VPN) overlays are normally used to connect multiples sites. Three approaches that share multiple features are VXLAN, NVGRE (Network Virtualization using Generic Routing Encapsulation) [9], and STT (Stateless Transport Tunneling) [10].

Comparing all previous possibilities, SPB and TRILL are not suitable to scale All-Path due to the partial functional overlapping with All-Path and the presence of other features not strictly needed by All-Path. This functional overlapping consists in the fact that SPB and TRILL solve the restrictions of the spanning tree protocol, but also All-Path does. Moreover, TRILL also overlaps in its application scenarios.

V. Conclusions

All-Path protocols had the problem of limited scalability. Although VXLANs are not optimal in overhead and latency, they provide a seamless extension of All-Path L2 domains and full interoperability with any IP compatible L2 domain: either RSTP/STP based or SDN domains. This fact greatly increases the scalability of networks using All-Path, SDN switches or other L2 protocols. A validation has been performed by implementing VXLANs on *OvS* (the *de facto* standard software switch), although other different scenarios and VXLANs variants are feasible. All-Path protocols are well-positioned future wise because they can be implemented in a variety of hardware and software (i.e. via P4 targets). All-Path protocols, extended with VXLANs, provide scalability, flexibility, simplicity and high performance in L2 domains.

Acknowledgment

This work was funded by Comunidad de Madrid through project TAPIR-CM S2018/TCS-4496 and grant PEJ-2018-AI/TIC-11803 and from University of Alcalá through "Programa de Formación del Profesorado Universitario (FPU)" and the project BISHOPS CCG2018_EXP-076.

References

[1] IEEE, "802.1aq - Shortest Path Bridging," 2006.
[2] R. Perlman and D. Eastlake, "Introduction to TRILL," *The Internet Protocol Journal*, vol. 14, no. 3, pp. 2–19, 2011.
[3] M. Mahalingam, D. Dutt, K. Duda, P. Agarwal, L. Kreeger, T. Sridhar, M. Bursell, and C. Wright, "Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks," RFC 7348, Aug. 2014.
[4] J. Alvarez-Horcajo, D. Lopez-Pajares, I. Martinez-Yelmo, J. Carral, and J. Arco, "Improving multipath routing of tcp flows by network exploration," *IEEE Access*, 2019.
[5] J. Alvarez-Horcajo, I. Martinez-Yelmo, E. Rojas, J. Carral, and D.Lopez-Pajares, "New cooperative mechanisms for software defined networks based on hybrid switches," *Transactions on Emerging Telecommunications Technologies*, vol. 28, no. 8, p. e3150, 2017.
[6] E. Fernandes and C. Rothenberg, "Openflow 1.3 software switch," *Salao de Ferramentas do XXXII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuıdos SBRC*, pp. 1021–1028, 2014.
[7] P. B. et al, "Onos: towards an open and distributed sdn os," in *Proceedings of the third workshop on Hot topics in software defined networking*. ACM, 2014, pp. 1–6.
[8] M. Alizadeh and et al, "CONGA: Distributed Congestion-aware Load Balancing for Datacenters," *SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 4, pp. 503–514, Aug. 2014.
[9] P. Garg and Y.-S. Wang, "NVGRE: Network Virtualization Using Generic Routing Encapsulation," RFC 7637, Sep. 2015.
[10] B. Davie and J. Gross, "A Stateless Transport Tunneling Protocol for Network Virtualization (STT)," Internet Engineering Task Force, Internet-Draft draft-davie-stt-08, Apr. 2016, work in Progress.