Rethinking IXPs' Architecture in the Age of SDN

Marc Bruyère, Gianni Antichi, Eder L Fernandes, Rémy Lapeyrade, Steve Uhlig, Philippe Owezarski, Andrew Moore, Ignacio Castro

Rethinking IXPs' Architecture in the Age of SDN

Marc Bruyere*, Gianni Antichi§, Eder L. Fernandes§, Remy Lapeyrade†,
Steve Uhlig§, Philippe Owezarski†, Andrew W. Moore‡, Ignacio Castro§
*Information Technology Center, University of Tokyo, JP ‡Computer Laboratory, University of Cambridge, UK
§Queen Mary University of London, UK †CNRS, LAAS, FR

***Abstract*—*Abstract*—Software Defined Internet eXchange Points (SDXs) are a promising solution to the long-standing limitations and problems of interdomain routing. While proposed SDX architectures have improved the scalability of the control plane, these solutions have ignored the underlying fabric upon which they should be deployed. In this paper, we present *Umbrella*, a software defined interconnection fabric that complements and enhances those architectures.**

Umbrella is a switching fabric architecture and management approach that improves the overall robustness, limiting control plane dependency and suitable for the topology of any existing Internet eXchange Point (IXP). We validate Umbrella through a real-world deployment on two production IXPs, TouSIX and NSPIXP-3, and demonstrate its use in practice, sharing our experience of the challenges faced.

I. Introduction

Internet eXchange Points (IXPs) are a central element of the Internet ecosystem: IXPs can carry huge traffic volumes and interconnect a multitude of networks of different types [1], [2]. The fundamental service provided by IXPs is a Layer-2 neutral facility where heterogeneous networks exchange IP traffic. While IXPs are the ideal vehicle to extend the benefits promised by Software Defined Networking (SDN) to the interdomain level [3], [4], [5], [6], reliability and scalability are essential aspects of an IXP that cannot be compromised by the introduction of SDN. Consequently, transforming IXPs from their legacy design into SDN-enabled fabrics is plagued with challenges. In particular, the impact of control channel disruptions or outages can cause severe disturbances, potentially affecting hundreds of networks and huge traffic volumes [7], [8], [9]. Moreover, control plane failures in large scale deployed SDN networks outweigh largely data or management ones combined together [10].

In this paper we propose *Umbrella*, a novel approach to IXP fabric management that can be deployed in any IXP topology to reduce the risks of a fabric excessively dependent on the control plane. Umbrella leverages SDN programmability to tackle in the data plane part of the control traffic, removing the actual MAC learning mechanism in legacy IXP networks. Specifically, the broadcasted Address Resolution Protocol (ARP) traffic is directly programmed within the data plane. Umbrella also uses a Layer-2 encoded path to minimize resource consumption, management cost and extend scalability. This approach greatly simplifies the management of the fabric: the only role of the controller is `supervising` the network by leveraging its global knowledge. Umbrella complements previous SDN architectures for IXPs in two manners. First, Umbrella supports the correspondingly enriched (and more complex) IXP through a more reliable and scalable fabric, in a similar fashion as [11] for iSDX [5]. Second, Umbrella's implementation supports SDN-enabled IXP architectures beyond the single hop IXPs (iSDX only allows single-hop fabrics), ensuring its applicability to any IXP topology. We envision SDN-enabled IXPs that enhance the controller's role as an intelligent supervisor, rather than an active and dangerously critical decision element. Umbrella is the first step in this direction.

The main contributions of this paper are:

- We introduce the Umbrella architecture and show how it leverages SDN programmability within the data plane.
- We show how Umbrella complements the current solutions for SDN-enabled IXPs [5] and allows their implementation in multi-hop IXPs, while reducing the risk of data plane disruptions.
- We present how to incrementally deploy Umbrella and demonstrate its practicality, by reporting on its deployment in two real IXPs: TouSIX and NSPIXP-3.
- We open source the Umbrella implementation[1], including the IXP-manager application that generates the Umbrella logic and BIRD (Bird Internet Routing Daemon) configuration for the Route Server (RS).

II. Background and Motivation

This section introduces IXPs' architecture, their main components –ARP-sponges and Route Servers (RSs)– and presents a motivating example for the proposed SDN-enabled fabric.

A. The IXP environment

IXPs are interconnection fabrics where multiple networks (i.e, members) meet to exchange traffic, frequently in huge volumes [12]. IXPs are typically implemented as a simple Layer-2 broadcast domain to which member networks connect BGP-speaking routers and exchange IP traffic. The network of an IXP is composed of edge and core switches [12]. The former connects to the IXP member routers while the latter interconnects physical IXP locations and aggregates the IXP traffic. IXP topologies range from single-hop cores, i.e., one core switch (e.g., AMS-IX, DE-CIX) to multiple-hops cores (e.g., LINX, MSK-IX).

Legacy IXPs typically operate at the Ethernet level and are rarely involved in routing decisions. To exchange traffic,

[1] http://github.com/umbrella-fabric/TouSIX-Manager

IXP members need to know each other physical address, i.e., MAC address, which they learn using the ARP protocol. However, ARP traffic in big IXP fabrics can be large enough to compromise low-capabilities routers [13]. Due to congestion during ARP storms, re-establishing BGP connections might be necessary, causing severe disturbances at the IXP which can even result in connectivity disruptions [7], [8]. The ARP traffic volume grows even higher during network outages [14], when many routers attempt to resolve the IP addresses of peers that are not available.

1) ARP-Sponge: The state-of-the-art solution to ARP storms is the ARP-Sponge[2]. An ARP-Sponge Server limits the ARP traffic if it exceeds a certain limit. When the number of ARP requests for an IP address reaches the threshold (e.g., because an interface is down an does not respond), the ARP-Sponge server *sponges* such IP address: the server replies with its own MAC address to the ARP requests of that node and from there on all the ARP traffic sent to that node is instead sent to the ARP Sponge server. When the ARP-Sponge server receives traffic from a sponged IP address, it ceases to sponge such IP address. Although the ARP-Sponge mechanism prevents the escalation of ARP traffic, it suffers from multiple limitations:

- Single point of failure: while several ARP-Sponge servers could run in parallel, this would add complexity.
- ARP Sponges do not eliminate all unwanted ARP traffic.
- To determine whether an IP address is reachable again, ARP-Sponges rely on heuristics that require broadcasting or flooding, which consumes routers' resources.
- The ARP-Sponge server might fail to notice that an interface is up again. The server periodically ARPs for *sponged* addresses, when an IP replies the server *unsponges* it. It may happen however that the device that replied comes back but the server fails to notice it.

2) Route Servers (RS): Originally, each BGP node connected to the fabric had to establish BGP peerings (using TCP connections) with every other IXP member to obtain information about the networks' prefixes reachable at the IXP and exchange traffic. As IXPs grew in size [15], this solution implied keeping too many BGP sessions, with the associated administrative overhead, operational burden, and the risk of pushing IXP members' routers to their limit. IXPs introduced `Route Servers` (RSs) to address this problem [16]. RSs store all the incoming route information from IXP members and forward it without modification to the other members. Thanks to the RS, an IXP member can receive all the routing information available at the IXP through a single BGP session. In particular, BIRD[3] is an open-source Internet routing daemon for UNIX-like platforms and the most popular RS at IXPs (e.g., LINX or DE-CIX) [16].

B. The case for a stronger control and data plane separation

Previous work has shown how OpenFlow (OF) [17] could be deployed at the exchange [18], [19], [4], [5] using an IXP fabric with a central controller for all the peering routers at the IXP. In such architecture, the SDN controller would be co-located with the RS to ensure that the SDN and BGP control planes can talk to each other with a minimal delay [4], [5]. Despite advantages such as richer policies, one challenge remains: data plane issues may affect control plane messages, leading to a slow or unresponsive control plane, further aggravating the effect on the data plane. The critical problem resides in the centralized ARP-proxy: delays in the control channel might lead to all the connection oriented mechanisms (i.e., BGP, TCP) failing. For example, if the ARP messages of a peering router suffer delays to reach the SDN controller inside the IXP fabric, all the BGP sessions between such router and its peers would also suffer, forcing (in the worst case scenario) establishing new connections. Note that while a distributed ARP proxy could alleviate some of this issues, OF does not support such feature.

We show in Fig. 1 the disruption caused by ARP delays on the data plane or BGP sessions. We emulated the above SDN scenario of a central controller co-located with the RS on Mininet [20]. We instantiated virtual containers acting as peering routers: one for the RS and two more working as client hosts directly connected to one peering router each. We represented the IXP as a single open vSwitch coupled with a Ryu [21] controller acting also as an ARP-proxy, as in [18], [19], [4], [5]. Fig. 1 shows that, even a small delay of a few tens of milliseconds for ARP messages may trigger much larger disruptions on the data plane or BGP. Given the large volumes of traffic and IXPs' critical role, such disruptions are not acceptable [7], [8]. To fully benefit from the advances brought by the existing literature [18], [19], [4], [5], we advocate a stronger separation between the control and the data plane. Our approach is one possible solution towards this stronger separation.

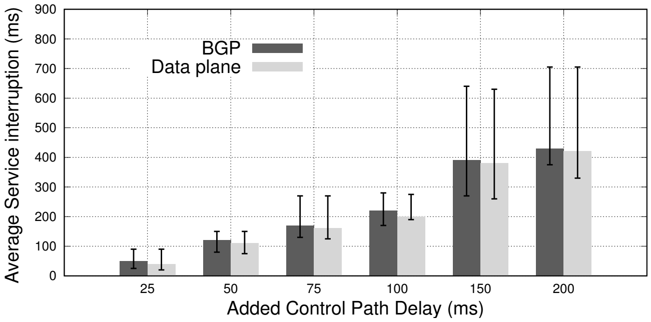


Fig. 1: The dependency between control and data plane.

III. A NEW SDN FABRIC FOR IXPS

We now present Umbrella: a new SDN fabric for IXPs, that focuses on the control-data plane dependencies to provide a robust, reliable, and scalable forwarding inside the IXP.

A. No broadcast traffic

IXPs apply strict rules [22], [23] to limit the side effects of a Layer-2 shared broadcast domain, e.g., the MAC address

[2] ARP-sponge manual: http://ams-ix.net/downloads/arpsponge

[3] http://bird.network.cz/

of the router with which the member connects to the IXP must be known in advance. Only then the IXP will allocate an ethernet port on the edge switch and configure a MAC filtering Access Control List (ACL) with that MAC address [24]. The location of all the member's routers is thus known to the IXP. Accordingly, Umbrella eliminates the need of location discovery mechanisms based on broadcast packets, i.e., ARP request or IPv6 neighbor discovery. Umbrella makes unnecessary the active ARP-proxy daemon proposed in previous SDN-enabled IXP solutions [18], [19], [4], [5]. Note that while [5] does not rely in broadcast-based discovery mechanisms, it still heavily depends on the ARP-proxy for mapping virtual to actual MAC and IP addresses. Umbrella makes on-the-fly translation of broadcast packets into unicast by exploiting the OF ability to rewrite the destination MAC address of a frame matching a rule [25].

We propose a label-oriented forwarding mechanism to reduce the number of rules inside the core of the IXP fabric. Umbrella edge switches explicitly write the per-hop port destinations into the destination MAC field of the packet. The first byte of the MAC address represents the output port to be used by the core switch. With Umbrella, the number of flow table entries per core switch will scale with the number of active physical ports in the switch itself. This is important to guarantee the fabric scalability. While Umbrella's encoding scheme is currently limited to 256 output ports per hop, more bits in the port encoding (thus mapping more physical ports) can be used.

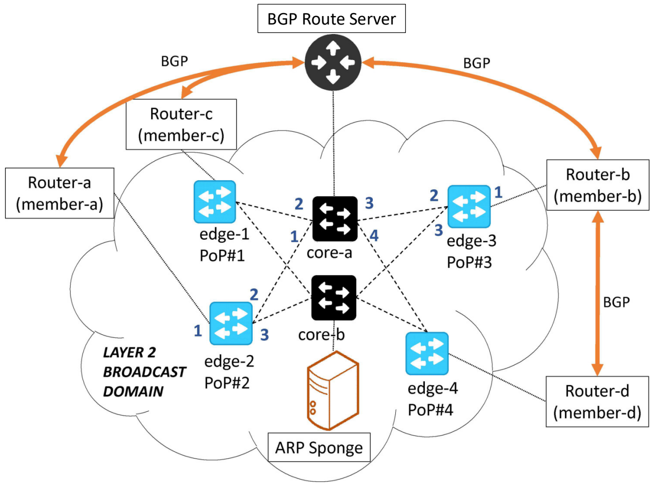


Fig. 2: Typical topology of a medium to large IXP.

We now explain how Umbrella works using the IXP topology in Fig. 2. The path to connect `router-a` to `router-b` through `core-a` traverses all the following ports: 1 and 2 in `edge-2`, 1 and 3 in `core-a` and 2 and 1 in `edge-3`. When the `router-a` sends an ARP request (i.e., broadcast message), the switch `edge-2` receives the frame, rewrites the destination MAC address with the corresponding output ports of the path, `03:01:00:00:00:00`, and forwards it to the core switch. Note that the output port of the first hop is not encoded: this port is directly set as an output action following the path encoding action.

Once the frame reaches the core (`core-a`), it is redirected to output port 3, and then to switch `edge-3` (i.e., the forwarding in the core is based on the most significant byte). Finally, `edge-3`, before forwarding the frame through the output port indicated in the second byte of the MAC address (output port 1), rewrites that field with the real MAC address of `router-b`.

When the source and destination are directly connected to the same edge switch, no encoding is needed, and the the edge switch directly replaces the broadcast destination address with the target MAC destination address. In an IPv6 scenario, the OF match pattern indicated in the edge switch needs to be on the IPv6 ND target field of the incoming ICMPv6 Neighbor Solicitation packet [26]. The matching table on the edge switch should maintain an association between IPv6 addresses and their location, as in the IPv4 case.

B. A label switching approach

Umbrella's forwarding mechanism allows reusing legacy switches in the core, limiting the burden (and costs) of upgrading the hardware. A core switch only needs to forward packets based on simple access filtering rules, whereas the edge switches need OF-like capabilities to rewrite the layer-2 destination field. While this approach is directly applicable to single-core IXP fabrics, it is not applicable to multiple-hops fabrics. With a single hop, the core switch would expect the output port to be encoded in the most significant byte of the destination MAC address. In the multi-hop case, since a packet can traverse multiple core switches, a new encoding scheme is needed to distinguish the output ports at different core switches. This is a fairly common case in hypercube-like topologies (e.g., LINX, MSK-IX).

Adapting Umbrella to multi-hop IXPs is far from trivial. An encoding of the Layer-2 destination address where the most significant byte refers to the output port of the first core switch, the second byte to the second switch, and so on, is infeasible: unfortunately, a core switch might be the first or the second on the path depending on the route. Another solution could be using the `input port` of the frame in the forwarding rules installed in the core switches. With the input port, it is possible to locate the switch on the path and therefore look at the correct byte in the Layer-2 destination address. Alas, this approach may not work in arbitrary topologies. Moreover, this mechanism will lead to a rule explosion in the core, as the number of forwarding entries grows quadratically with the number of possible input ports. Instead, to deal with multi-hop IXPs, Umbrella leverages source routing in the following manner. Initially, the first edge switch selects the path. Then, an ordered list of `output ports` is encoded in the destination MAC address as a stack of labels. Finally, each core node processes the frame according to the value on the top of the stack and pops it before forwarding the frame. With this configuration each switch only needs to look at the most significant byte of the address, regardless of its position in the path toward the destination. Popping out from the MAC destination address, the last label used requires header rewriting capabilities, making this solution feasible only for

OF-enabled core switches. In particular, every core switch must have two action tables: forwarding and copy-field[4].

C. Umbrella and Route Servers

Umbrella handles the forwarding of BGP traffic for both the usual bilateral peerings and RSs. For bilateral BGP sessions, the TCP connection is treated as pure data plane traffic crossing the IXP, and the traffic is handled by the switch rules, without any control plane intervention. For the RSs, the BGP traffic entering the fabric is directed to the RS with a single rule at the edge switch, while the egress traffic is handled automatically through the existing rules on the edge switches.

D. Failure detection and recovery

Umbrella relies on well known OF features. In particular, Group Fast Failover is the OF 1.1 mechanism to react to link failures [27]. A fast failover group table can be configured to monitor the status of ports and interfaces and the switch forwarding actions, independently of the controller. Recovering from a data plane failure is more challenging. In this scenario an active probing of the data plane status from the controller is needed (for a detailed discussion see [28]). In particular, the Umbrella controller can be instructed to implement the Local Link Discovery Protocol (LLDP) [27] or the Bidirectional Forwarding Detection (BFD) protocol [29]. Once a data plane failure has been detected, the Umbrella controller changes only the edge switch configuration with a fallback path.

IV. Experimental Evaluation

In this section, we evaluate the Umbrella architecture. We first estimate the average number of flow rules needed at the edge if Umbrella was to be implemented at different large Layer-2 neutral IXPs (Section IV-A). Then we calculate the impact of the ARP proxy on the data plane performance and compare it with the Umbrella approach.

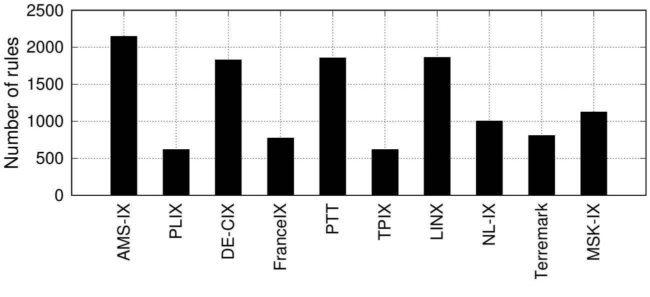


Fig. 3: Average number of rules required at the edge.

A. Flow table occupancy at the edge

Mapping the broadcast destination MAC address to the path inside the fabric requires additional flow rules at the edge of the fabric. This section provides a scalability analysis in terms of the required number of rules. From an ingress traffic point of view, Umbrella requires the OF switches at the edge to store three entries per peering router connected to the fabric. This is a necessary and sufficient condition to enable routing in any situation (i.e., IPv4 ARP, IPv6 Neighbor Solicitation and data plane traffic). All three entries share a common action in the rewriting and forwarding process due to the common treatment of Umbrella for location discovery and data plane traffic. From an egress traffic point of view, each edge switch needs to rewrite the MAC destination field of the received frame with the correct target. As the value to be inserted in the destination MAC field depends on the output port of the edge switch, the number of rules depends on the number of peering routers connected to the edge switch. The total number of rules for an edge switch is then the sum of the ingress and the egress ones.

Fig. 3 shows the average number of rules per edge switch if Umbrella was to be implemented at different large layer-2 neutral IXPs. As the number of rules also depends on how many peering routers are connected to the fabric, we rely on PeeringDB to asses this extent [30], [31].

The Umbrella architecture is applicable in today's IXPs, as shown by the relatively low number of flow entries required on Fig. 3. Indeed, today's OF-enabled switches already support flows ranging from a few thousands (e.g., Pica8 switches) to hundreds of thousands (e.g., Corsa and Noviflow switches)[5].

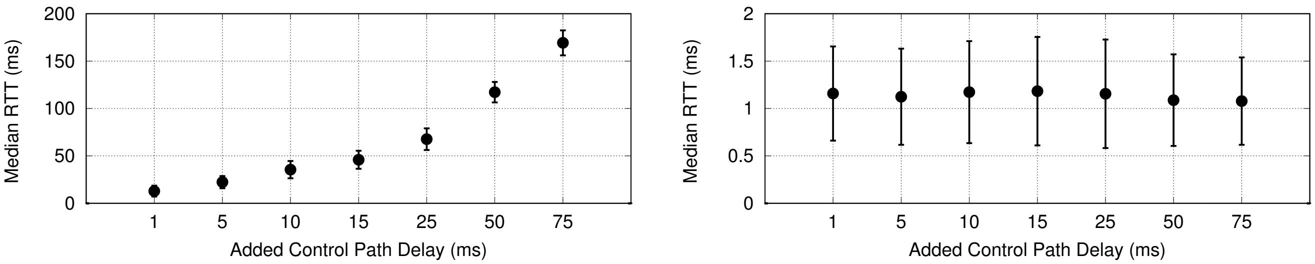
B. Impact of control channel on the data plane performance

To estimate the impact of an unreliable control channel on the data plane performance, we compared Umbrella with the ARP-proxy application in a realistic scenario following a three-pronged approach. First, we studied the effect over the Round Trip Time (RTT) of ARPs (i.e., the time required to receive the ARP reply after an ARP request is sent) under artificially induced delays. Secondly, we carried a similar analysis by studying the effects of packet loss. Finally, we evaluated the impact of Umbrella and the ARP-proxy on the fabric's throughput when IXP members send traffic to their peering partners.

1) Experimental set-up: To demonstrate the advantages of Umbrella and the drawbacks of approaches that rely on the control plane for implementing location discovery mechanisms, we reproduced an SDN IXP with 100 members and a RS controller managed either by Umbrella or an ARP-proxy application. The ARP-proxy and Umbrella are implemented as applications on top of the Ryu controller. Using Mininet [20], we emulated an IXP with a ring topology composed of three Open vSwitches, 102 Quagga BGP routers and one RS This setup mirrors the real TouSIX topology, though with more members. While this provides a representative benchmark, unfortunately, emulating larger scale networks is too resource intensive. In this topology, each peer privately peers with a single customer, and $1/3$ of the participants peer openly at the RS. Accordingly, there is a total of 88 simultaneous BGP sessions in the network. Larger numbers of peering sessions at the RS resulted in very slow BGP convergence due to the high number of prefixes exchanged. The generation of ARP

[4] OF 1.5 specifications allow copying and rewriting part of a header field.

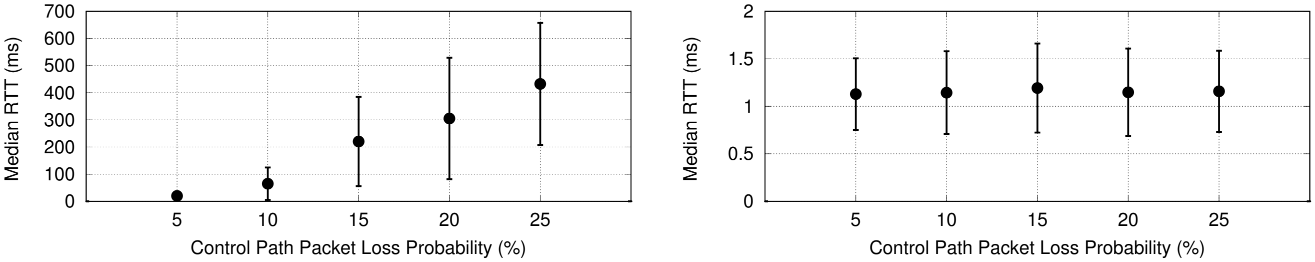
[5] See http://pica8.org/blogs/?p=565, http://www.corsa.com/sdn-done-right/ and http://noviflow.com, respectively.



(a) ARP-proxy

(b) Umbrella

Fig. 4: Median RTT of ARP packets when the control channel suffers different delays (whiskers indicate the median +/- MAD).



(a) ARP-proxy

(b) Umbrella

Fig. 5: Median RTT of ARP packets when the control channel suffers different packet loss rates (whiskers indicate the median +/- MAD).

traffic in our experiments is like in a real IXP: ARP requests are triggered when a BGP connection is started and requests are issued until a reply is received. When the IXP members exchange traffic ARP messages are also sent across them. The experiments were performed in a machine with 16 Intel(R) Xeon(R) CPUs @ 3.00GHz and 32GB of RAM.

2) Impact of delays in the control channel: We now show in Fig. 4a the median RTT for the pair of all ARP requests/replies when the control channel experiences different delays. To give a better idea of the distribution, the whiskers indicate the median RTT +/- the Median Absolute Deviation (MAD). The plot shows that delays in the control plane translates into increasing RTTs of ARPs, which result in delayed BGP connections, ultimately undermining the ability of peers to exchange through the fabric.

For Umbrella, the case is rather different. Fig. 4b shows that, as expected by design, the control plane does not affect the ARP packets' RTTs. With Umbrella, the ARP requests are sent directly through the fabric without interaction with the central controller. The broadcast traffic is converted into unicast traffic and directly forwarded to the destination. It also eliminates the typical flooding in layer 2 networks when a switch has not learned yet the port associated to a MAC address, therefore avoiding bandwidth wasting.

3) Impact of packet loss on the control channel: Here, we study the impact of losses on the control channel on the RTT of the ARP packets. Fig. 5a and 5b show the median RTT, in both the ARP-proxy approach and Umbrella, respectively.

Fig. 5b stresses the strong separation between the control and data plane in Umbrella: there is almost absolute insensitivity between the packet loss rate on the control channel and the RTT of the ARP packets. In Fig. 5a, we show how the ARP-proxy solution again suffers more and more as the loss rate on the control plane increases. The ARP-proxy solution suffers because the TCP connection between the OF-enabled switch and the controller is sensitive to packet losses, hence notably increasing the response time to the ARP requests.

4) Impact on the data plane throughput: We now evaluate the potential impact of the control plane on the data plane throughput inside of the IXP fabric. Using the same setup from the previous experiments, peers now generate TCP packets using the iperf tool[6] and exchange traffic with the others routers present in the fabric. There are 88 iperf sessions, equal to the number of peering sessions. For this experiment, the links between the routers of the members and the switches of the IXP are limited to 50Mbps, as large traffic loads could overwhelm the emulator and result into a bottleneck. Consequently, the maximum throughput that could be observed is 4400Mbps. Differently from the previous two experiments, we introduce no added loss or delay to the control plane, instead we focus on the throughput that an Umbrella vs. and ARP-proxy enabled IXP can deliver.

Fig. 6 depicts the median throughput delivered by the IXP fabric for 10 trials, each executed for a period of 10 minutes. The whiskers indicate the median +/- the MAD. Because of the slow start of TCP, in the initial seconds of the experiment the throughput is low both for Umbrella and the ARP-proxy. As the experiment progresses and the TCP sessions increase their congestion window, Umbrella delivers a higher throughput than the ARP-proxy during the whole experiment. Furthermore, the upper whisker shows that Umbrella nearly reaches the maximum possible throughput. Considering the high level of performance that IXPs are expected to achieve, the results

[6] https://iperf.fr/

clearly show the advantages of Umbrella over control plane dependent solutions.

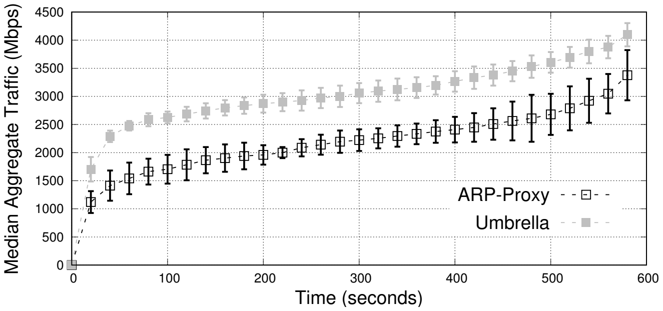


Fig. 6: Median throughput in an IXP controlled by an ARP-proxy or Umbrella (whiskers indicate the median +/- MAD).

V. Experience with real deployments

In this section, we show our effort in bringing the research outcomes from Umbrella into operation. Specifically, we show how Umbrella has been introduced in TouSIX and NSPIXP-3 OSAKA.

Toulouse IXP. TouSIX is a non-profit neutral IXP organization founded in 2005 (Toulouse, France), providing an interconnected network infrastructure with four Points of Presence (PoPs) and connected with the FranceIX (Paris) and LyonIX (Lyon) IXPs. From May 2015 it is the first European IXP to fully leverage OF for its day-to-day operations. The TouSIX architecture is based on our Umbrella approach, providing us with operational experience of a real-world deployment.

The management of the legacy IXP was a complex task with problems resulting from wrong configurations done by the members (i.e., unwanted loops) and several broadcast storms. The lack of an appropriate monitoring infrastructure as well as the irregular incidence of these problems made the detection of these events hard. Umbrella was implemented to overcome these issues as well as to improve the stability and manageability of the whole infrastructure.

The new TouSIX OF-enabled topology has three PoPs with one OF-switch. Each PoP acts as an edge and there are no core switches. The three switches have been programmed according to the Umbrella principles and replace the previous legacy switches. A long testing period was required to validate the system configuration and its stability before going into production, with the first test campaign dating December 2014. At that time, only Pica8 switches had full hardware integration with the Open vSwitch agent, making the Pica8 P-3290 switches the natural choice. Unfortunately, the latest PicOS version available during the initial part of the tests did not allow installing rules matching the ipv6_nd_target field. This option is necessary to enable the Umbrella scheme for IPv6 traffic (see Section IV-A). In addition, the software was flushing the entire flow table during hardware reboots or when the connection to the controller was lost: close collaboration with Pica8 solved these issues, triggering a new PicOS version (2.6).

The new OF-enabled fabric went live in May 2015. The OF Pica8 switches are placed on the top of the existing Cisco switches. While the former are being used for the data plane traffic, the latter have been kept to transport the control plane traffic. Thanks to the new switches, the transmission bandwidth has been improved by one order of magnitude on 2 out of 3 connections between edge switches. Member cables have migrated from the legacy Cisco switches to Pica8, with only a few seconds of service interruption per member. The Pica8 switches have been configured to not flush the flow table when the OF Agent looses connectivity with the OF controller. The deployment leverages Ryu [21], the NTT Labs open-source controller, while parallel development is underway for the ON.Labs' ONOS controller [32].

The Umbrella TouSIX fabric has been running seamlessly and with a reduced dependence on the network administrator. There is no more broadcast ARP traffic (see Table I), freeing the 10 members from receiving undesired traffic while allowing them to announce 399 IP prefixes. Umbrella reduced the total average volume of ARP traffic flowing through the fabric by 97%. Note that Umbrella solves here an additional problem: with this topology, a simple layer-2 approach combined with an ARP-proxy and an ARP-sponge would require of a spanning tree protocol to avoid loops.

TABLE I: ARP traffic (packets/second), in the legacy & Umbrella IXPs at TouSIX & NSPIXP-3 Osaka.

ARP traffic		TouSIX		NSPIXP-3	
		Legacy	Umbrella	Legacy	Umbrella
Max	(Pkt/s)	14.96	3	20	2
Average	(Pkt/s)	8.51	0.18	14	0.5
Min.	(Pkt/s)	1.1	0	12	0

TABLE II: Peak & average traffic (MegaBits per second) at TouSIX & NSPIXP-3 Osaka swicthes.

Total traffic volume	TouIX	NSPIXP-3
Peak (Mb/s)	9767	748
Average (Mb/s)	48	229

Table II shows the peak and average traffic of TouSIX and NSPIXP-3. TouSIX has a total of 14 nodes connected, including both routers and servers. The switches must include rules for the MAC destination forwarding, ARP target routing, and for the ICMPv6 ND target routing. As only IPv4 is on production at this stage, the total number of rules per switch is 28. The Umbrella design also frees capacity in the switches that have a CPU utilization with virtually no spikes, stable around the 8%. Differently, the TouIX switches did suffer spikes of CPU usage reaching the 100% of their capacity, which resulted from issues related to the convergence of different control plane protocols.

In terms of recovery time, when a switch with an empty table connects to the controller, installing the corresponding flow rules (i.e., the database polls and send rules to the switch) takes about 5.6s. and a hard reboot is about 64s. Note that due to the design of Umbrella, connecting a new router to TouSIX is a "plug & play" operation: the administrator automatically installs the corresponding flow rules for any "approved" router because the controller already knows its configuration. Consequently the time since an approved router is connected and it is fully operative is negligible. On the contrary, if a router is connected without approval of the administrator, such router will see all its traffic dropped. As the TouSIX-Manager operates the topology with proactive procedures, if the link in the control path is lost, all the rules already deployed on any switch remain. We tested the effect of turning off the TouSIX manager, to verify that traffic statistics where as usual without impacting the fabric operations.

NSPIXP-3 OSAKA. Launched by the WIDE project in 1997, NSPIXP-3 is the oldest IXP in Osaka (Japan). NSPIXP-3 has a single site, 10 nodes connected, and uses one switch with two VLANs, one for local peering and a second VLAN to interconnect the Osaka and Tokyo domain. In July 2017 a pre-deployment of Umbrella in combination with Faucet started. FAUCET [33] is a very compact open source OF controller, enabling network operators to run networks the same way they do server clusters. The final migration to production has been achieved in December 2017. This deployment confirms the real-world applicability of Umbrella and demonstrates its success in almost eliminating the ARP traffic (see Table I).

VI. Related works

Introducing OF at the IXP world is a recent idea, [4], [5] developed a SDN-based eXchange point (SDX) to enable more expressive policies than legacy solutions while scaling to hundreds of participants while achieving sub-second convergence in response to configuration changes and routing updates. To achieve this, the multi-table version of the iSDX prototype considers a scenario where all the participants are connected through a single switch. In reality, however, there might be multiple hops within the IXP. Umbrella and iSDX are complementary designs: Umbrella, can support the iSDX architecture by directly forwarding the location discovery packets to each participant. In particular, after the decision process for the egress port performed by iSDX, the destination MAC address can be encoded using Umbrella. The packet is then delivered following the path encoded in the destination MAC. However, the proposed integration would require an additional OF switch to send ARP requests, for virtual next hops that would be handled by an ARP proxy. As Umbrella turns every broadcast ARP into unicast, the ARP requests can have their destination encoded within the path to the ARP Proxy. This would enhance flexibility, because the ARP proxy does not need to be connected to a specific switch in the IXP fabric[7].

ENDEAVOUR [11] leverages the Umbrella scheme to obtain an effective transport layer over a multi-hop topology, by removing all the side effects associated to broadcast traffic, and by enabling efficient access control over the fabric.

The Cardigan project [3] implements a hardware based, default deny policy, capable of restricting traffic based on RPKI verification of routes advertised by devices connected to layer3 fabric. While this approach offers the required protection for a stable IXP, it is less suitable for IXPs that wish to remain neutral with regards to IP forwarding.

The ONOS CASToR [6], with its *ARP Hygiene* also relies on unicasting broadcast packets. However, the primary CASToR objective is to provide flexibility to operators to interconnect through a User Interface and APIs. CASToR can face scalability limitations as they use a single table.

While label switching techniques in combination with source routing has been used in the past, MAC-based routing in OF networks is a fairly new [34], [35]. [36] show that the destination MAC address can be used as a universal label in SDN environments and the ARP caches of hosts can be exploited as an ingress label table, shrinking the forwarding tables of network devices. [37] demonstrate that, using destination MAC addresses as opaque forwarding labels, an SDN controller can leverage large MAC forwarding tables to manage a plethora of fine-grained paths. Although these approaches have very nice properties for large-scale networks, they still rely on an ARP-proxy mechanism, which involves the limitations already discussed.

VII. Conclusion

Umbrella enhances IXP reliability, manageability and scalability. By handling the control traffic directly within the data plane, Umbrella reduces failures/disruptions and complements existing IXP architectures, enabling the deployability of existing SDX architectures in any IXP. We demonstrate the scalability of Umbrella practical and its applicability by reporting on two successful deployment in real IXPs. We see Umbrella as a first step towards SDN architectures less dependent on the control plane, supporting the controller in its role of an intelligent supervisor, rather than as an active and dangerously critical decision point.

Acknowledgements

We thank J. Bailey (Google), C. Combes, H. Brunel and L. Guerby (TouSIX), S. Garrison (Pica8), M. Pels and Ariën Vijn (AMS-IX) for their very valuable inputs. This work is supported by the EARL EPSRC project (EP/P025374/1).

References

[1] B. Ager, N. Chatzis, A. Feldmann, N. Sarrar, S. Uhlig, and W. Willinger, “Anatomy of a Large European IXP,” in *SIGCOMM*. ACM, 2012.

[2] N. Chatzis, G. Smaragdakis, J. Böttger, T. Krenc, A. Feldmann, and W. Willinger, “On the Benefits of Using a Large IXP as an Internet Vantage Point,” in *IMC*. ACM, 2013.

[3] J. Stringer, D. Pemberton, Q. Fu, C. Lorier, R. Nelson, J. Bailey, C. Correa, and C. Esteve Rothemberg, “Cardigan: SDN distributed routing fabric going live at an Internet exchange,” in *ISCC*. IEEE, 2014.

[4] A. Gupta, L. Vanbever, M. Hahbaz, S. Donovan, B. Schlinker, N. Feamster, J. Rexford, S. Shenker, R. Clark, and E. Katz-Bassett, “SDX: A Software Defined Internet Exchange,” in *SIGCOMM*. ACM, 2014.

[5] A. Gupta, R. MacDavid, R. Birkner, M. Canini, N. Feamster, J. Rexford, and L. Vanbever, “iSDX: An Industrial-Scale Software Defined Internet Exchange Point,” in *NSDI*. USENIX, 2016.

[6] H. Kumar, C. Russell, V. Sivaraman, and S. Banerjee, “A software-defined flexible inter-domain interconnect using onos,” in *EWSDN*. IEEE, 2016.

[7] H. D. Vu and J. But, “How RTT Between the Control and Data Plane on a SDN Network Impacts on the Perceived Performance,” in *ITNAC*. IEEE, 2015.

[7] The ARP proxy function in iSDX is essential to compress the flows into Forwarding Equivalent Classes.

[8] K. He, J. Khalid, A. Gember-Jacobson, S. Das, C. Prakash, A. Akella, L. E. Li, and M. Thottan, "Measuring Control Plane Latency in SDN-enabled Switches," in *SOSR*. ACM, 2015.

[9] V. Giotsas, C. Dietzel, G. Smaragdakis, A. Feldmann, A. Berger, and E. Aben, "Detecting Peering Infrastructure Outages in the Wild," in *SIGCOMM*. ACM, 2017.

[10] R. Govindan, I. Minei, M. Kallahalla, B. Koley, and A. Vahdat, "Evolve or Die: High-Availability Design Principles Drawn from Googles Network Infrastructure," in *SIGCOMM*. ACM, 2016.

[11] G. Antichi, I. Castro, M. Chiesa, E. L. Fernandes, R. Lapeyrade, D. Kopp, J. H. Han, M. Bruyere, C. Dietzel, M. Gusat, A. W. Moore, P. Owezarski, S. Uhlig, and M. Canini, "ENDEAVOUR: A Scalable SDN Architecture for Real-World IXPs," *JSAC*, vol. 35, no. 11, 2017.

[12] B. Ager, N. Chatzis, A. Feldmann, N. Sarrar, S. Uhlig, and W. Willinger, "Anatomy of a Large European IXP," in *SIGCOMM*. ACM, 2012.

[13] M. Wessel and N. Sijm, "Effects of IPv4 and IPv6 address resolution on AMS-IX and the ARP Sponge," Master's thesis, Universiteit van Amsterdam, the Netherlands, 2009.

[14] "FranceIX outage," https://www.franceix.net/en/events-and-news/news/franceix-outage-notification/, [Online; accessed Feb. 2018].

[15] J. C. Cardona and R. Stanojevic, "IXP Traffic: A Macroscopic View," in *LANC*. ACM, 2012.

[16] P. Richter, G. Smaragdakis, A. Feldmann, N. Chatzis, J. Boettger, and W. Willinger, "Peering at Peerings: On the Role of IXP Route Servers," in *IMC*. ACM, 2014.

[17] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: Enabling Innovation in Campus Networks," *CCR*, vol. 38, no. 2, 2008.

[18] V. Boteanu and H. Bagheri, "Minimizing ARP traffic in the AMS-IX switching platform using OpenFlow," Master's thesis, Universiteit van Amsterdam, the Netherlands, 2013.

[19] I. Pepelnjak, "Could IXPs Use OpenFlow to Scale?" MENOG, 2012.

[20] N. Handigol, B. Heller, V. Jeyakumar, B. Lantz, and N. McKeown, "Reproducible network experiments using container-based emulation," in *CoNEXT*. ACM, 2012.

[21] "Ryu controller," http://osrg.github.io/ryu, [Online; accessed Feb. 2018].

[22] AMS-IX, "Allowed Traffic Types," http://ams-ix.net/technical/specifications-descriptions/allowed-traffic, [Online; accessed Feb. 2018].

[23] M. Hughes, M. Pels, and H. Michl, "Internet Exchange Point Wishlist," https://www.euro-ix.net/en/forixps/ixp-wishlist/, 2015, [Online; accessed Feb. 2018].

[24] Open-IX, "IXP Technical Requirements," http://www.open-ix.org/cpages/ixp-technical-requirements, [Online; accessed Feb. 2018].

[25] "OpenFlow Switch Specification," https://www.opennetworking.org/wp-content/uploads/2013/04/openflow-spec-v1.0.0.pdf, [Online; accessed Feb. 2018].

[26] T. Narten, E. Nordmark, W. Simpson, and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)," Internet Requests for Comments, RFC 4861, September 2007.

[27] S. Sharma, D. Staessens, D. Colle, M. Pickavet, and P. Demeester, "Enabling Fast Failure Recovery in OpenFlow Networks," in *DRCN*. IEEE, 2011.

[28] D. Kreutz, F. M. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2015.

[29] N. L. V. Adrichem, B. J. V. Asten, and F. a. Kuipers, "Fast Recovery in Software-Defined Networks," *EWSDN*, 2014.

[30] "PeeringDB," http://www.peeringdb.com, [Online; accessed Feb. 2018].

[31] A. Lodhi, N. Larson, A. Dhamdhere, C. Dovrolis, and k. Claffy, "Using peeringDB to Understand the Peering Ecosystem," *CCR*, vol. 44, no. 2, 2014.

[32] "ONOS," http://onosproject.org, [Online; accessed Feb. 2018].

[33] J. Bailey and S. Stuart, "Faucet: Deploying SDN in the Enterprise," *Communications of ACM*, vol. 60, no. 1, 2016.

[34] A. Hari, T. Lakshman, and G. Wilfong, "Path switching: Reduced-state flow handling in SDN using path information," in *CoNEXT*. ACM, 2015.

[35] K. Agarwal, C. Dixon, E. Rozner, and J. Carter, "Shadow MACs: Scalable Label-switching for Commodity Ethernet," in *HotSDN*. ACM, 2014.

[36] A. Schwabe and K. Holger, "Using MAC Addresses As Efficient Routing Labels in Data Centers," in *HotSDN*. ACM, 2014.

[37] K. Agarwal, C. Dixon, E. Rozner, and J. Carter, "Shadow MACs: Scalable Label-switching for Commodity Ethernet," in *HotSDN*. ACM, 2014.

Marc Bruyere started his career in 1996 working for Club-Internet.fr, and for Cisco, Vivendi Universal, Credit Suisse First Boston, Airbus/Dimension Data, Force10 Networks, and Dell. He received is Ph.D degree from the LAAS CNRS, his thesis is about Open Source OpenFlow SDN for IXPs. He designed and deployed the first European OpenFlow IXP fabric for the TouIX. Now he is a PostDoc at the University of Tokyo.

Gianni Antichi received a Ph.D. degree in information engineering from the University of Pisa (2011). He is currently a Lecturer at Queen Mary University of London, UK. His research spans the area of reconfigurable hardware, high speed network measurements and Software Defined Networking.

Eder L. Fernandes received a MSc degree from the Universidade of Campinas (UNICAMP) in 2014 and worked in industry during four years on Research and Development of SDN tools. He is currently a Research Assistant and PhD candidate at Queen Mary University of London. His research interests include network measurements, SDN, routing, and network simulation.

Remy Lapeyrade received his engineering diploma in computer networks and telecommunications from UPSSITECH, France. He is currently pursuing his Ph.D.in LAAS-CNRS, France. His research is focused on improving scalability, resiliency and security of IXP networks using SDN technologies.

Steve Uhlig received a Ph.D. degree in applied sciences from the University of Louvain (2004). He is currently a Professor of Networks at Queen Mary University of London. His research interests are focused on the large-scale behaviour of the Internet, Internet measurements, software-defined networking, and content delivery.

Philippe Owezarski is director of research at CNRS (the French center for scientific research), working at the Laboratory for Analysis and Architecture of Systems, Toulouse, France. He received a Ph.D. in computer science in 1996 from Paul Sabatier University, Toulouse III, and his habilitation for advising research in 2006. His main interests deal with next generation Internet, more specifically taking advantage of IP networks monitoring and machine learning for enforcing quality of service and security.

Andrew W. Moore the reader in systems at the University of Cambridge, U.K., where he jointly leads the Systems Research Group working on issues of network and computer architecture with a particular interest in latency-reduction. His research contributions include enabling open network research and education using the NetFPGA platform; other research pursuits include low-power energy-aware networking, and novel network and systems data center architectures.

Ignacio Castro is currently a Research Associate at Queen Mary University of London. He received his Ph.D. degree while researching at the Institute IMDEA Networks. His research interests focus on the macroscopic behaviour of the Internet, the economics of network's interconnections, Internet measurements and Software Defined Networking.