P4IX: A Concept for P4 Programmable Data Planes at IXPs

Daniel Wagner
DE-CIX
MPI-INF
Frankfurt, Germany

Matthias Wichtlhuber
DE-CIX
Frankfurt, Germany

Christoph Dietzel
DE-CIX
MPI-INF
Frankfurt, Germany

Jeremias Blendin
Intel, Barefoot Switch Division
Santa Clara, United States of America

Anja Feldmann
MPI-INF
Saarland Informatic Campus
Saarbücken, Germany

ABSTRACT

Internet Exchange Points (IXPs) are a crucial part of the Internet's infrastructure. Large IXPs can potentially interconnect thousands of ASes and facilitate the exchange of more than 10 Tbps of traffic during peaks. However, their specific technical requirements (e.g., large Layer-2 domains, complex traffic filtering) are not well addressed by today's networking hardware, as vendors optimize for the ISP market due revenues that are orders of magnitude higher. Software Defined internet eXchanges (SDXes) are a promising solution since they enable tailored hardware and software stacks to satisfy the specific IXP requirements. They combine a high degree of automation with the flexibility to implement value-added services and, thus, may reduce IXP's costs. Since previous work is based on the OpenFlow standard, which was last updated in 2017, we revisit the idea by leveraging the flexibility of P4 networking hardware. We present the P4IX, a technical concept for a generic P4 packet processing pipeline for IXPs. The P4IX concept is built upon a comprehensive requirements analysis: we characterize the IXP landscape and provide first-hand insights of a large IXP operator (more than 1000 well distributed ports). Moreover, we use our insights to critically discuss the P4IX from an operational, technical, and organizational perspective.

CCS CONCEPTS

• **Networks** → **Programmable networks**; **Public Internet**; *Network design principles*; *Wide area networks*.

KEYWORDS

SDN, SDX, IXP, P4

ACM Reference Format:
Daniel Wagner, Matthias Wichtlhuber, Christoph Dietzel, Jeremias Blendin, and Anja Feldmann. 2022. P4IX: A Concept for P4 Programmable Data Planes at IXPs. In *ACM SIGCOMM 2022 Workshop on Future of Internet Routing & Addressing (FIRA '22), August 22, 2022, Amsterdam, Netherlands.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3527974.3545725

FIRA '22, August 22, 2022, Amsterdam, Netherlands

ACM ISBN 978-1-4503-9328-7/22/08…$15.00
https://doi.org/10.1145/3527974.3545725

1 INTRODUCTION

Internet eXchange Points (IXPs) are traffic hubs between Autonomous Systems (ASes) and facilitate the settlement free exchange of traffic over a Layer-2 platform (peering) [7]. The largest IXPs ensure low-latency interconnection for hundreds or even thousand ASes and exchange 10 Tbit/s or more during peak times[1]. The Software Defined internet eXchange (SDX) concept, first introduced by Gupta et al. [18], shows how Software Defined Networking (SDN) can benefit Internet Exchange Points (IXPs) [1]. The basic idea of an SDX is to tailor a soft-/hardware stack to the requirements of IXPs by relying on SDN building blocks. Such specific requirements include the need to realize one large *Layer-2* domain with the inherent broadcast problems, e.g., for ARP, as well as sophisticated inbound and outbound traffic filtering, while providing the reliability expected from critical infrastructures.

The body of SDX works [6, 8, 16–18, 22, 23, 26] relies on the OpenFlow paradigm [24]. In 2017, the OpenFlow standard was augmented by the Open Networking Foundation with the increased capabilities and flexibilities of a P4-enabled stack [4, 5]. P4 is a domain-specific language which defines how packets are processed by the data plane, i.e., switches or routers. The language allows the definition of custom packet header parsing and assembly as well as match/action pipelines to perform non-trivial operations on packets in line rate. To the best of our knowledge, there has been no work on a holistic SDX concept that takes advantage of the P4 capabilities[2]. Thus, we revisit the question of how to realize a P4IX. Our motivation is two-fold: (a) we have first-hand experience from operating a very large distributed IXP (more than 1,000 ports across many data centers), which allows us to precisely scope P4IX requirements and (b) we find that OpenFlow's limitations have lead to a number of non-optimal design choices.

Firstly, we review some of the limitations of using OpenFlow: (a) previous solutions enabled multi-hop IXPs by using MAC headers for encoding routing information (VMAC concept) [2, 6, 18]—this implies a loss of compatibility to the existing Layer-2 switching paradigms and complicates debugging. Rather, this should be realized in the data plane and an external controller should only be required when the set of IXP members changes or additional hardware is added or removed. Indeed, relying on an external control for IXPs is complex as they are critical infrastructure and, thus, require

[1] E.g., AMS-IX (https://stats.ams-ix.net/index.html), DE-CIX (https://de-cix.net/en/locations/frankfurt/statistics), and LINX (https://portal.linx.net/okta-login).
[2] Silva et al. [11, 12] focus coping with elephant flows in IXP networks.

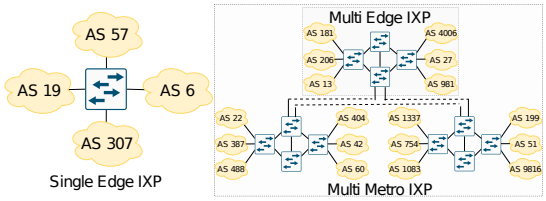


Figure 1: Different IXP setups ranging from a single edge IXP to an interconnected multiple metropolitan area IXP. The setup implements a Layer-2 domain even for multi metro IXPs.

a fail-safe complex controller setup (e.g., Martins et al. [23] uses a distributed ONOS controller). (b) Mechanisms implementing fast rerouting on link failure are notoriously hard to solve using OpenFlow only, since they either require large state space on the switches or involve the OpenFlow controller [2, 6, 18] for rerouting, which introduces latency and packet loss. Katta et al. [20, 21], in a data center context, show how to tackle this problem using P4 in the data plane, a solution we incorporate in our concept. Additionally, our approach is transparent and does not require changes in the member networks, in contrast to member operated SDN controllers [26].

We use our experience from operating a large distributed IXP to outline how to take advantage of P4-enabled hardware for a P4IX. We conceptualize a P4IX in the context of realistic technical, operational, and business requirements of IXPs. It provides a solid basis for adding more sophisticated services such as (a) BGP policy enforcement in the data plane [16], (b) data plane integrated DDoS mitigation [13, 27], and (c) other value-added services as outlined in [8]. More precisely, we make the following contribution:

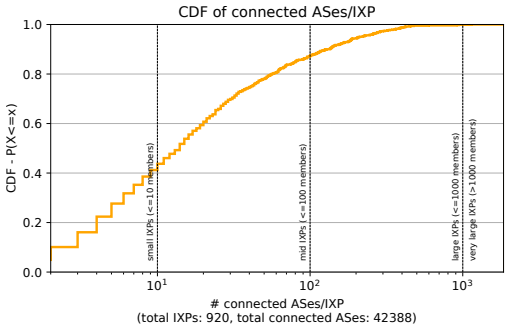
- A characterization of the IXP landscape/market and implications for P4IXes and an overview of operational and technical peculiarities of IXPs.
- An outline of technical and operational P4IX requirements and a P4IX pipeline concept.
- A critical discussion of operational, technical, and organizational P4IX advantages and disadvantages.

2 IXP LANDSCAPE CHARACTERIZATION

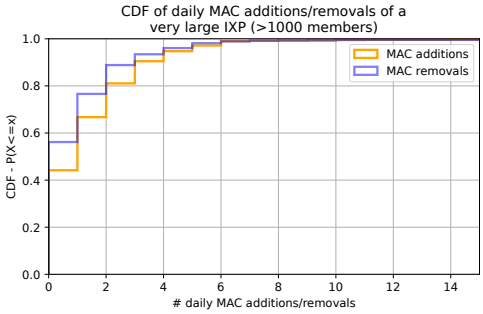
In this section we touch upon IXP market specific properties and their implications for a P4IX. We start by looking at the IXP landscape using data from PeeringDB[3] —a central database for IXPs and peerings in general. Figure 2a plots a CDF of the number of ASes that are members at each IXP. As such, the x-axis shows the log-scaled number of connected ASes per IXP and the y-axis the fraction of IXPs with that number of member ASes. Already this simple figure provides a number of relevant insights.

The network hardware market size of IXPs is negligible compared to the one of ISPs and Cloud providers. 40% of all IXPs have less than ten member ASes, 80% have less than 50 member ASes, see Figure 2a. In addition, given that there are 920 IXPs in the PeeringDB that interconnect 42,303 ASes, we get a ratio of 46 ASes per IXP. Even if we assume that each IXP member AS operates only one router–the edge router visible to the IXP–this implies that the hardware footprint of IXPs is at least 46 times smaller compared to ISP/Cloud provider ASes that they interconnect. In practice, this

[3]https://www.peeringdb.com, last accessed 11/24/2021.



(a) CDF of connected ASes per IXP, data from peeringDB.com, 11/24/2021.



(b) Daily MAC additions/removals in the peering LAN of a very large IXP from 07/2017-11/2021.

Figure 2: IXP landscape

gap is even orders of magnitude larger. Thus, since the IXP networking hardware market segment is small, hardware vendors tend to tailor solutions for ISPs instead of IXPs. However, IXPs' network architectures differ substantially from ISPs'. While ISPs predominantly operate Layer-3 architectures, IXPs operate large Layer-2 architectures.

IXP scalability requirements. On the one hand, the majority of IXPs can at least theoretically be realized on a small hardware platform, i.e., a single 32 port switch, if we ignore redundancy requirements, and a 64 port switch if we keep a redundant port for each member (single edge IXPs, see Figure 1). On the other hand, large and very large IXPs have a share of 15% of the overall IXP market. Their setups need to provide connectivity for hundreds to thousands of ports. Moreover, they are often distributed across multiple data centers in a geographical region (multi edge IXP, see Figure 1). Very large IXPs, e.g., LINX, AMS-IX, and DE-CIX, operate separate IXP locations around the globe which are interconnected by a global backbone network (multi metro IXP, see Figure 1).

As such P4IX platforms have to be scalable from very small setups to very large ones. Moreover, they need to support low-end hardware as well as high-end hardware. Ideally, a P4IX platform supports scale out: if an IXP requires more ports, additional switches can be added, thus eliminating the need to replace existing hardware.

IXP business challenges. Acquiring additional member ASes is becoming increasingly difficult for IXPs. The pool of potential non-peering ASes is shrinking as peering has become a common practice

over the last decade. Moreover, transit traffic prices have been declining for over a decade[4]. This, in turn, applies pressure on peering prices, as both are to some extent competing products for the same traffic. In the foreseeable future, IXPs have the following options to generate growth: (a) geographical expansion/internationalization, i.e., establishing new IXP locations; (b) expanding to new customer segments beyond ISPs, CDNs, and Cloud providers; (c) providing additional services on top of peering to increase members' added value and, thus, the IXP's revenue. The geographical expansion strategy underlines the P4IX scalability requirements. Addressing new customer segments and implementing value-added services require (a) a high degree of automation and (b) the ability to realize proprietary services on top of the P4IX platform.

IXP operational challenges. The large asymmetry in member ASes among IXPs implies a large asymmetry in revenue and, thus, in resources available for research and development of a P4IX architecture. Even very large IXPs are no multi-billion dollar enterprises. They often operate under non-profit membership governance models. It is unlikely that any single IXP can amortize the cost for developing its own P4IX platform. Consequently, an open source community effort is needed. Still, at the same time there is a need for service differentiation. Thus, the base open source P4IX platform has to be able to support custom P4IX services on top to foster competition and innovation.

IXP networks evolve slowly. While the IXP community has introduced a common API for members to adjust configurations at the IXPs[5], some human interactions between the IXP operator and the member ASes are still required. Consequently, IXP networks evolve slowly and constitute a mostly static network environment. To underline this slow rate of change Figure 2b shows a CDF of the daily MAC address changes at a very large IXP over the last 4 years. On about 50% of days, no MAC addresses were added or removed; on 99% of days fewer than 8 MACs were changed. Since any MAC change results in a routing change, this corresponds to the number of required routing changes due to members joining/leaving the IXP. A P4IX architecture can take advantage of this mostly static setup to simplify operations, by, e.g., replacing difficult-to-administer routing protocols at IXPs with multiple switches with offline optimized static routes.

3 CHALLENGES WHEN REALIZING LARGE IXPS USING ISP HARDWARE

Before discussing how to build a P4IX, we next outline the technical challenges of realizing very large IXPs with traditional ISP-focused hardware.

Large IXPs are victims of feature bloat. Traffic rates at large IXPs easily reach an average of multiple terabits per second. At the same time, IXPs are typically built using a comparably small number of hardware units (if compared to ISPs). Thus, they require compact hardware to reach the required port density at a reasonable energy consumption. These requirements make off-the-shelf hardware, in particular data center switches, inadequate. Operators of large IXPs are thus left to build their platforms using high-end (expensive) service routers tailored for ISP requirements. These routers offer a massive feature set, e.g., they can be used as Broadband Remote Access Servers (BRAS) that terminate a large number of end customer subscriber lines, while offering services such as admission control, traffic shaping, and integration into billing infrastructures. However, IXPs only need a very small subset of these features, i.e., mainly Layer-2 functionality such as switching and VLANs, but have to pay the surplus for all implemented features.

Unsuitable Feature Set Realization. Many feature designs and implementations are targeted for the ISP market rather than the IXP one due to its relatively small size. This starts with the standardization phase where the standardization committees are driven by ISPs and hardware vendors, as IXPs often lack resources to participate in all processes. One recent example is BGP FlowSpec, a method to communicate traffic filters via BGP from, e.g., an ISP customer to an ISP. It is "primarily designed to allow an AS to perform inbound filtering in their ingress routers of traffic that a given downstream AS wishes to drop." [19]. As such, hardware vendors implement FlowSpec filtering on the ingress path. In an IXP setting, this is unfeasible as IXPs handle peering traffic and tend to filter on egress [14] to protect member ports. Other examples include: statistics export limited to Layer-3/4, IXPs need Layer-2 information additionally; limited ingress/egress filtering on Layer-2 or Layer-3/4 exclusively, IXPs need Layer-2 and Layer-3/4 filtering to ensure platform stability and to implement value-added services.

Challenges of large Layer-2 domains. IXPs are realized via large Layer-2 domains. This comes with a number of challenges unknown to ISPs. These include broadcasting based protocols, like ARP and IPv6 neighborhood discovery (NDP). Due to the large number of peers, large volumes of broadcast traffic can accumulate. This can overwhelm member routers, e.g., with information on MAC to IP mappings of routers from ASes that they do not peer with. Thus, Broadcast, Unknown unicast, and Multicast traffic (BUM) should be dropped or at least rate limited in peering LANs.

Another challenge are Layer-2 loops—if two IXP ports are short-circuited in a way allowing Ethernet frames to be forwarded back into the IXP platform. Frames may be forwarded in an endless loop, which leads to overwhelmed backplanes in the edge switches and thus affects neighboring member ports on the same switch. Since Ethernet frames do not carry a time to live counter, an efficient detection is difficult. Therefore, it is necessary to filter which MAC address can send frames from which port and to block other traffic.

When an IXP grows to a multi-site setup, it needs routing—a Layer-3 feature—while maintaining the facade of a single Layer-2 domain. Often routing is realized via OSPF or IS-IS over MPLS. To maintain the Layer-2 domain and enable traffic engineering over the now present Layer-3, virtualization features are used, including VPLS, EVPN, and ECMP. This often goes along with using a vendor proprietary network management system. The setup introduces unnecessary cost and complexity—full routing is realized even though the setup is more or less static and all paths (including redundant ones) are known upfront, members join or leave only occasionally and their routers have well-known static MAC and IP addresses behind their dedicated port.

[4] https://drpeering.net/FAQ/What-are-the-historical-transit-pricing-trends.php, data up to 2015, last visited 29/11/2021; the industry is not publishing transit/peering prices, so the absolute values have to be taken with a grain of salt. However, the trend is clear.
[5] Standardized IX-API (https://ix-api.net/)

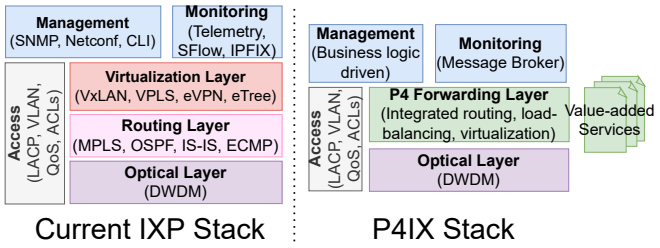


Figure 3: Traditional IXP setup versus P4IX concept.

4 P4IX REQUIREMENTS

Next, we outline requirements for a P4IX based on our experience operating a very large IXP. We divide them into operational and technical ones and assign them an identifier, i.e., technical requirement "TR1" and operational requirement "OC1". We later refer to these identifiers in the P4 pipeline concept (see Figure 4).

Among the overarching requirements are: (a) avoiding unnecessary complexity and implementing the bare minimum functionality for operating IXPs ranging from small single-edge ones to large multi-metro ones and (b) offering enough flexibility to implement value-added services. Today's services include dropping of unwanted traffic (blackholing [14]) and isolation via virtualization, e.g., peering with Cloud providers via dedicated peering LANs. Regarding future services enabled by P4IX, we point to the work of Chiesa et al. [8]. These may include DDoS mitigation, enforcement of BGP policies (e.g., IRR/RPKI checks) in the data plane, and application specific peerings (e.g., video). All these services rely on the following building blocks: virtualization, admission control, traffic classification and filtering. Thus, P4IXes need to support these for a wide variety of packet header fields (Layer-2 to Layer-4).

P4IX Technical requirements. Firstly, Layer-2 traffic engineering capabilities similar to a WAN with MPLS or ECMP are required (TR1) for both single-metro and multi-metro setups (similar to [20, 21, 25]). However, we note that routes are almost static. Supporting multiple virtual peering LANs requires virtualization functionalities such as VLANs (TR2). Fine grained inspection of packet header fields is required to classify traffic, e.g., to avoid broadcast traffic, handle Layer-2 loops and BUM traffic (TR3). Note, TR2-3 are also needed to realize value-added services. ARP and NDP remain the protocols to ensure Layer-2 (Ethernet) connectivity. Thus, an IXP specific ARP/NDP handling mechanism which avoids broadcast storms is required (TR4). To enable members to purchase port capacities independent of their physical access bandwidths, rate limiting and Link Aggregation Groups (LAGs) are required (TR5). To enable monitoring and to generate statistical summaries, packet sampling is required (TR6). Lastly, no data plane software update should interrupt forwarding of traffic (TR7).

P4IX Operational requirements. Here, we see a trade-off between degree of automation vs. capability of manual intervention for Network Operation Center (NOC) teams. The former is desirable to take full advantage of reducing operational cost. The latter may be required to troubleshoot problems. Thus, a P4IX concept requires a tight integration with IXP business logic, e.g., when a port is sold by the IXP provider, it should be automatically provisioned (OR1), but at the same time, there should be tooling to manually override automatic processes to be able to fix operational problems (OR2). For troubleshooting, monitoring information should be exposed to external tools (OR3).

Stage	Key	Value
Ingress & Egress Rate Limiter	Physical Port, MAC, VLAN	Bandwidth
Ingress & Egress MAC/IP Filter	Layer-2, Layer-3, Layer-4 Headers	Drop or pass
ARP / NDP Handler	IPv4 / IPv6-Address	MAC-Address
Dst. Classification	MAC-Address	Port Group

Table 1: Lookup table contents of the P4 pipeline stages.

5 THE P4IX: TECHNICAL CONCEPT

This section presents a technical concept for a P4IX. The main idea, see Figure 3, is to merge the virtualization and routing layer into a single P4 forwarding layer that relies on static routing. This reduces complexity, i.e., by removing all routing protocols. Furthermore, the P4IX concept does not introduce any changes to the optical underlay network, nor the platform's access technologies. Next, we discuss our concept in a bottom-up fashion.

5.1 P4 Forwarding Layer

Figure 4 outlines the P4 forwarding layer (bottom) together with its management layer (top) required for operation. The main interface

They interact via P4 lookup tables—the main interface for configuring the P4 pipeline. These are filled by the management layer and, then, used by the P4 pipeline at runtime.

We show and discuss the P4 pipeline from left to right. Packets enter the pipeline through the ingress port (*Input from port*) and leave through the output port (*Output to port*). When a packet enters the P4 pipeline, its headers have to be parsed (*Input from port*). A P4 parser graph is used to parse Ehternet, VLAN, and IPv4/v6 headers for further processing. Packets with invalid headers are detected by the parser and dropped immediately. This stage also separates header and payload, where the header is passed and transformed along the pipeline, while the payload stays until both are merged to be emitted via the output port (*Output to port*).

The stage *Ingress Rate Limiter* is responsible for rate limiting member traffic on ingress to the IXP's infrastructure based on a table that contains a mapping of ingress port, MAC, and VLAN tag to purchased capacity.

Next, the packet is matched against ingress filters to drop or rate limit traffic (*Ingress MAC/IP Filter*). This stage enforces basic Layer-2 network security by, e.g., tying each port to a single router MAC source address. It handles VLAN admission by matching source and destination MAC addresses as well as the VLAN tag against a table with whitelisting rules provided by the management layer. Any non-matching packets are dropped or shaped as required. Moreover, this stage is used to realize value-added services such as member or application specific peering LANs.

Packets passing the *Ingress MAC/IP Filter* are subject to a *Traffic Classification* stage. Here, each packet is tagged with one of three classes: (a) ARP / NDP traffic, (b) Broadcast, unknown Unicast or Multicast traffic (BUM), or (c) other traffic. This classification is then used to pass the packets to their next stage: *ARP / NDP Handler*, *BUM Handler*, or directly to the *Dst. Classification*. The *ARP / NDP Handler* is an IXP specific implementation for address resolution

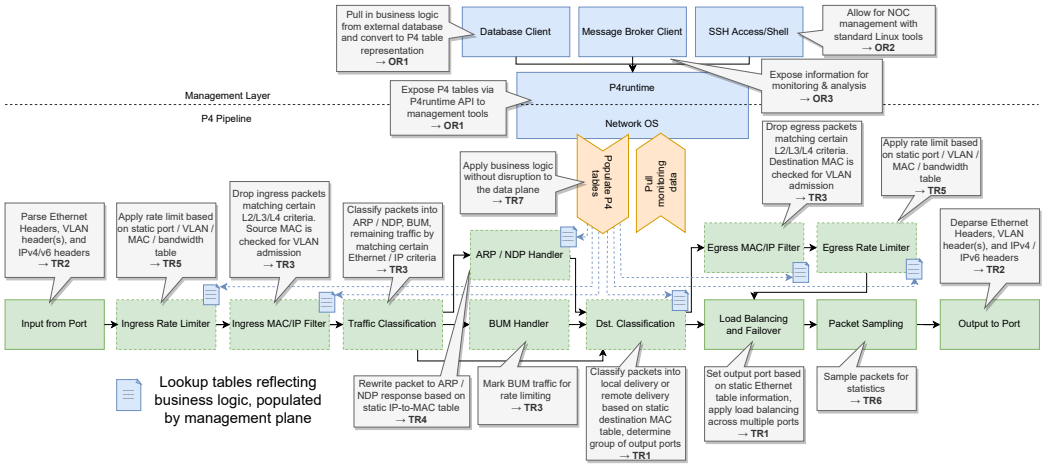


Figure 4: The P4 IXP model. The blue boxes indicate management-related solutions, green boxes show the parts of the P4 forwarding layer. Dashed boxes are not required for forwarding in the core, all boxes are required for edge routers.

protocols. ARP and NDP requests are immediately transformed into replies using a MAC to IP mapping table generated by the management layer. This is feasible due to the relative static nature of IXP networks. This prevents ARP and NDP requests from flooding the IXP internal network. The *BUM Handler* rate limits BUM traffic to avoid BUM packets overwhelming the platform.

The next step in the pipeline determines the packet's rough destination (*Dst. Classification*). Two scenarios are possible: (a) *local* delivery, i.e., the destination interface is the current edge switch or (b) *remote* delivery, i.e., the packet needs to be routed to another edge switch to reach its destination. As all members' MAC addresses are known in advance, this can be realized via local lookup tables.

Next, packets marked for *local* delivery are sent to the *Egress MAC/IP Filter* stage. This stage matches headers of various layers against egress filtering rules. These rules can be used to reflect user-defined blackholing/DDoS filtering measures (for reasons why this should be done on egress, see [14]). The next stage in this branch, the *Egress Rate Limiter* marks packets to be dropped if the member's rate limit is exceeded.

The *local* and *remote* paths merge again in the *Load Balancing and Failover* stage. However, they are handled separately. For *remote* delivery, there are typically multiple links available (due to redundancy requirements) for entering the IXP's core. Thus, this stage is responsible for distributing the packets among the available links according to metrics defined by the IXP operator, enabling, e.g., cost based routing or ECMP. In case of *local* delivery, packets are forwarded to the member while respecting LAGs. This stage is responsible for realizing failover mechanisms using, e.g., in-data-plane probing [21, 25], based on the management's layer holistic view of the network topology.

The *Packet Sampling* stage exports data for statistical and monitoring purposes. This data is traditionally generated from sampled packets, aggregated on a per-flow basis. In our pipeline, any stage that decides to drop a packet will mark this in the packet's metadata. It then continues to travel through the P4 pipeline until the egress parser stage. Thus, the sampling stage has full visibility of all packets entering the pipeline even if they are dropped, including the reason why the packet is dropped. This, as well as the full visibility of all Layer-2 to Layer-4 information, offers better visibility of IXP relevant data. Note, most traditional packet sampling protocols only export Layer-3/4 headers.

The final stage reads the packet metadata, deparses the respective headers for either remote or local delivery. If it is not marked drop it reassembles the packet and emits it via the output port.

5.2 Management and Monitoring

The above pipeline is based on pre-populated lookup tables, visualized as blue boxes in Figure 4, that reflect the IXP's functionality, i.e., ranging from member specific rate limits, LAG or VLAN configurations, to blackholing rules, as well as internal routing decisions and value-added service specific information. Thus, we next discuss options for how to populate these tables.

- The *traditional approach* is the most wide-spread management approach in today's networks. It implements embedded management and control on the device by providing CLI access and a limited set of configuration protocols (e.g., SNMP). The CLI is usually proprietary. Hardware vendors often provide an additional, proprietary network management system (NMS) that abstracts the CLI details under a GUI. This approach comes with vendor lock-in and due to its proprietary nature often leads to challenging automation.
- The *model-exposure approach* works by exposing a formal model of valid networking hardware configurations to the outside world. This allows validation of configurations or configuration changes upfront, which is highly beneficial for automation. After validation, changes are transferred to the hardware and applied to the data plane, usually in a transaction-based manner enabling rollbacks upon deployment failure. Netconf/YANG [3, 15] provide a standardized method that is currently reaching operative environments.
- The *external controller* approach was made popular by OpenFlow [4]. The networking hardware outsources all logic to external controller software, which instructs the hardware using a narrow interface to push forwarding decisions. All management functionality is implemented in the controller. This approach is very flexible, but also introduces complexity

by adding a potentially large controller stack, which needs to be fail-safe for critical infrastructure to the system.

From the perspective of an IXP operator, mixing the traditional and the model-exposure approach is most promising since the external controller approach is a risky choice. The model-exposure approach allows for a tight integration with the IXP's business logic. It can be used to pull IXP member data from external data sources like ERP systems, generate configuration changes, validate them, and apply them to the data plane. This software-driven process enables a high degree of automation. At the same time, IXP NOC teams need to be able to troubleshoot networking hardware. These teams usually do not have software development skills and should, therefore, remain able to override any configuration change manually using a CLI until problems are fixed in the software stack. This aligns well with today's operational practices.

The model-exposure approach is achieved through two key software components that run on the host system of the P4 switch: (a) a database client that is connected to an external database containing all information related to the IXP business logic and (b) a data transformation pipeline executed upon every update of the database. The external database is not to be confused with a traditional SDN controller, as the P4 switch remains fully operational even if the database fails. The intelligence resides in the data transformation pipeline that generates P4 lookup table contents according to the IXP business logic and passes them to the P4 runtime, which in turn populates the P4 tables in the data plane accordingly. This is depicted with blue icons and dashed arrows in Figure 4. For a list of what information is stored in the lookup tables, we refer to Table 1.

Besides management, monitoring plays an important role. Currently, a large zoo of protocols is used, e.g., for the export of sampled packet data, there are three competing standards: SFlow, Netflow, IPFix [9, 10, 28]; other monitoring information such as interface information is exported via SNMP or Streaming Telemetry as promoted by OpenConfig[6]. Moreover, monitoring data needs to be distributed to a larger number of endpoints with different purposes, e.g., billing, network monitoring, statistics, DDoS mitigation, and data warehouses. The existing approaches are not well-suited for this scenario, as they are designed for a small number of data sinks. Consequently, often message brokers take on this job, i.e., there is a central infrastructure (e.g., a Kafka cluster) tasked with distributing measurement and monitoring data to different endpoints after decoding the necessary information from its wire format. With a P4IX architecture, this step can be skipped and a message broker client can run directly on the switch operating system (e.g., Linux) and push the necessary information directly to the message broker—notwithstanding that a standardized format is used for messaging to remain compatible with existing monitoring tools.

6 DISCUSSION

The proposed P4IX is superior to the earlier proposed OpenFlow based SDX realizations (e.g., no reliance on external controllers, solutions for link failure, etc.). As such, it offers many advantages for IXP operators; however, moving an IXP to P4IX imposes significant changes, which introduces new challenges as well. In the following, we discuss advantages and disadvantages of the P4IX.

[6]https://www.openconfig.net/, last visited 12/1/2021

P4IX Advantages. The P4IX is based on standardized P4 which is supported by a wider range of hardware. Thus, it prevents vendor lock-in, which promises a considerable reduction in capital expenditures as well as a larger range of hardware. This is a considerable advantage given the highly varying requirements of various IXPs. In addition, the P4IX should reduce operational expenditures as the network becomes easier to manage and operate. On the one hand, technical complexity, e.g., routing protocols, can be eliminated, while on the other hand, tightly coupling the IXP's business logic with the data plane simplifies provisioning of new members and running value-added services. Moreover, it is now possible for IXPs to implement new functionalities themselves or tailor existing functionality to IXP-specific challenges. These can then be rolled out immediately. IXPs, thus, no longer need to wait until traditional hardware vendors release a requested feature. This enables tailored, potentially highly customized solutions independent from the rigid feature set or hardware constraints of ISP-oriented hardware. This promises to reduce the time-to-market, e.g., for value-added services. Moreover, the P4 implementations can be made open source, to fuel community driven collaboration between IXPs. IXPs can also share their implementations with their members so that these can review code and increase their trust in the platform's software.

P4IX Challenges. While a P4IX is promising, it does not come for free. A considerable shift in the IXP's development and management is required. So far, data plane development and, in particular, software development has not been the focus of IXP staff. This has been the responsibility of the hardware vendor. The same applies for data plane testing regarding compatibility with other hardware and for operational stability. Thus, we argue that, due to the size and structure of IXPs, data plane development and testing should be done collaboratively based on an open source foundation. While open source *control plane* projects such as the BIRD route server [29] are providing critical services at IXPs, *data plane* projects have not yet been set up. Moreover, to test the stability and compatibility of a production-ready P4IX is likely to require a substantial test lab.

Moreover, the hardware packaging of current P4 switches is not yet a good match for medium to large IXPs due to their port density requirement, i.e., for connecting hundreds to thousands of member ASes. Additional hardware is required to aggregate these links before being connected to the IXP's edge. This generates the need for additional rack space and increases the energy consumption. However, large and very large IXPs are the ones that may drive the development of a P4IX—they are the ones with the resources.

Besides the technical challenges, a P4IX also imposes organizational challenges. A higher degree of automation such as proposed in this work requires the retraining of the network engineers and NOC members at the IXP. In the mid-term, the CLI-oriented mindset of IXP staff needs to be transformed into a mindset organized around software development teams, i.e., release cycles, software engineering methods like SCRUM and development sprints.

Acknowledgements

We thank the anonymous reviewers for their constructive comments. We further thank the DE-CIX staff for their ongoing support. This work was partially funded by the German Federal Ministry of Education and Research (BMBF) grant "5G-INSEL" (16KIS0691) and "Souverän. Digital. Vernetzt." Joint project 6G-RIC (16KISK027).

REFERENCES

[1] B. Ager, N. Chatzis, A. Feldmann, N. Sarrar, S. Uhlig, and W. Willinger. Anatomy of a large european ixp. In *Proceedings of the ACM SIGCOMM 2012 conference on Applications, technologies, architectures, and protocols for computer communication*, pages 163–174, 2012.

[2] 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. *IEEE Journal on Selected Areas in Communications*, 35(11):2553–2562, Nov 2017.

[3] M. Bjorklund. The YANG 1.1 Data Modeling Language. Request for Comments 7950, Internet Engineering Task Force, 2016.

[4] P. Bosshart, D. Daly, G. Gibb, M. Izzard, N. McKeown, J. Rexford, C. Schlesinger, D. Talayco, A. Vahdat, G. Varghese, et al. P4: Programming protocol-independent packet processors. *ACM SIGCOMM Computer Communication Review*, 44(3):87–95, 2014.

[5] P. Bosshart, G. Gibb, H.-S. Kim, G. Varghese, N. McKeown, M. Izzard, F. Mujica, and M. Horowitz. Forwarding metamorphosis: Fast programmable match-action processing in hardware for sdn. *SIGCOMM Comput. Commun. Rev.*, 43(4):99–110, Aug. 2013.

[6] M. Bruyere, G. Antichi, E. L. Fernandes, R. Lapeyrade, S. Uhlig, P. Owezarski, A. W. Moore, and I. Castro. Rethinking ixps' architecture in the age of sdn. *IEEE Journal on Selected Areas in Communications*, pages 1–1, 2018.

[7] N. Chatzis, G. Smaragdakis, A. Feldmann, and W. Willinger. There is more to ixps than meets the eye. *ACM SIGCOMM Computer Communication Review*, 43(5):19–28, 2013.

[8] M. Chiesa, C. Dietzel, G. Antichi, M. Bruyere, I. Castro, M. Gusat, T. King, A. W. Moore, T. D. Nguyen, P. Owezarski, et al. Inter-domain networking innovation on steroids: Empowering ixps with sdn capabilities. *IEEE Communications Magazine*, 54(10):102–108, 2016.

[9] B. Claise. Cisco Systems NetFlow Services Export Version 9. Request for Comments 3954, Internet Engineering Task Force, 2004.

[10] B. Claise and B. Trammell. Information Model for IP Flow Information Export (IPFIX). Request for Comments 7012, Internet Engineering Task Force, 2013.

[11] M. V. B. da Silva, A. S. Jacobs, R. J. Pfitscher, and L. Z. Granville. Ideafix: identifying elephant flows in p4-based ixp networks. In *2018 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6. IEEE, 2018.

[12] M. V. B. da Silva, A. S. Jacobs, R. J. Pfitscher, and L. Z. Granville. Predicting elephant flows in internet exchange point programmable networks. In *International Conference on Advanced Information Networking and Applications*, pages 485–497. Springer, 2019.

[13] A. da Silveira Ilha, Â. C. Lapolli, J. A. Marques, and L. P. Gaspary. Euclid: A fully in-network, p4-based approach for real-time ddos attack detection and mitigation. *IEEE Transactions on Network and Service Management*, 2020.

[14] C. Dietzel, M. Wichtlhuber, G. Smaragdakis, and A. Feldmann. Stellar: Network attack mitigation using advanced blackholing. In *Proceedings of the 14th International Conference on Emerging Networking EXperiments and Technologies*, CoNEXT '18, pages 152–164, New York, NY, USA, 2018. ACM.

[15] R. Enns, M. Bjorklund, J. Schoenwaelder, and A. Biermann. Network Configuration Protocol (NETCONF). Request for Comments 6241, Internet Engineering Task Force, 2011.

[16] A. Gupta, N. Feamster, and L. Vanbever. Authorizing network control at software defined internet exchange points. In *Proceedings of the Symposium on SDN Research*, pages 1–6, 2016.

[17] A. Gupta, R. MacDavid, R. Birkner, M. Canini, N. Feamster, J. Rexford, and L. Vanbever. An industrial-scale software defined internet exchange point. In *2016 USENIX Annual Technical Conference (USENIX ATC 16)*, Denver, CO, 2016. USENIX Association.

[18] A. Gupta, L. Vanbever, M. Shahbaz, S. P. Donovan, B. Schlinker, N. Feamster, J. Rexford, S. Shenker, R. Clark, and E. Katz-Bassett. Sdx: A software defined internet exchange. In *Proceedings of the 2014 ACM Conference on SIGCOMM*, SIGCOMM '14, pages 551–562, New York, NY, USA, 2014. ACM.

[19] S. Hares, R. Raszuk, D. McPherson, and M. Bacher. Dissemination of Flow Specification Rules. Request for Comments 8955, Internet Engineering Task Force, 2020.

[20] N. Katta, A. Ghag, M. Hira, I. Keslassy, A. Bergman, C. Kim, and J. Rexford. Clove: Congestion-aware load balancing at the virtual edge. In *Proceedings of the 13th International Conference on Emerging Networking EXperiments and Technologies*, CoNEXT '17, pages 323–335, New York, NY, USA, 2017. ACM.

[21] N. Katta, M. Hira, C. Kim, A. Sivaraman, and J. Rexford. Hula: Scalable load balancing using programmable data planes. In *Proceedings of the Symposium on SDN Research*, SOSR '16, pages 10:1–10:12, New York, NY, USA, 2016. ACM.

[22] L. A. D. Knob, R. P. Esteves, L. Z. Granville, and L. M. R. Tarouco. Mitigating elephant flows in sdn-based ixp networks. In *2017 IEEE Symposium on Computers and Communications (ISCC)*, pages 1352–1359. IEEE, 2017.

[23] L. F. C. Martins, Í. Cunha, and D. Guedes. An sdn-based framework for managing internet exchange points. In *2018 IEEE Symposium on Computers and Communications (ISCC)*, pages 00996–01001. IEEE, 2018.

[24] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner. Openflow: enabling innovation in campus networks. *ACM SIGCOMM computer communication review*, 38(2):69–74, 2008.

[25] R. Miao, H. Zeng, C. Kim, J. Lee, and M. Yu. Silkroad: Making stateful layer-4 load balancing fast and cheap using switching asics. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, pages 15–28, 2017.

[26] H. Mostafaei, D. Kumar, G. Lospoto, M. Chiesa, and G. Di Battista. Desi: A decentralized software-defined network architecture for internet exchange points. *IEEE Transactions on Network Science and Engineering*, 2021.

[27] F. Musumeci, A. C. Fidanci, F. Paolucci, F. Cugini, and M. Tornatore. Machine-learning-enabled ddos attacks detection in p4 programmable networks. *Journal of Network and Systems Management*, 30(1):1–27, 2022.

[28] P.Phaal, S. Panchen, and N. McKee. InMon Corporation's sFlow: A Method for Monitoring Traffic in Switched and Routed Networks. Request for Comments 3176, Internet Engineering Task Force, 2001.

[29] P. Richter, G. Smaragdakis, A. Feldmann, N. Chatzis, J. Boettger, and W. Willinger. Peering at peerings: On the role of ixp route servers. In *Proceedings of the 2014 Conference on Internet Measurement Conference*, IMC '14, pages 31–44, New York, NY, USA, 2014. ACM.