HolistIX: a zero-touch approach for IXPs

Christoff Visser[1], Seiichi Yamamoto[2], Tomine Takashi [3], Yuji Sekiya[2], and Marc Bruyere[1.4]

[1]IIJ Innovation Institute, Japan
[2]The University of Tokyo, Japan
[3]National institute of information and communications, Japan
[4]Japanese-French Laboratory of Informatics, CNRS IRL3527, Japan

***Abstract*—Zero-Touch is a broad concept, but it is understood as a means to limit human exposure to the data-plane states, and therefore reduces the potential for error. By combining the various layers from the administration level down to the switching fabric, introducing a new top to bottom architecture.**

This paper presents our operational research project HolistIX, a full-stack management solution enabling zero-touch networking within IXP infrastructure networks. HolistIX achieves this by first bringing diagramming to the forefront of topology planning and automated network configuration generation. This is then followed by verifying the proposed network configuration through emulation before deployment to the production network. By utilizing Umbrella SDN switching fabric, HolistIX enables full-stack management, with minimal human interaction required.

We also briefly present the results of deploying our HolistIX approach in two IXPs, one in France (TouIX) and the other in Japan (DIX-IE).

I. Introduction

A principal objective of zero-touch is to use software-based solutions to automate some well-understood tasks for time and cost savings. Nowadays, the Internet reflects an economically globalized society, and zero-touch is more critical than ever in our day-to-day practice. Internet eXchange Points (IXP) are points of interconnect where IXP peer participants, also known as members, come together to exchange traffic amongst themselves for mutual benefit. IXPs are a fundamental aspect of the Internet as they allow for lower latency, higher bandwidth and add a layer of redundancy for a lower cost when compared to purchasing transit traffic from global transit providers. In the paper, we use the term *member(s)* to refer to the networks and organization peering at the IXP and *operator* as the IXP organization and infrastructure.

IXPs provide an interconnect [1] between members using a common Layer 2 protocol, usually Ethernet, through a switching fabric. Border Gateway Protocol (BGP) [2] is commonly run on top of the fabric to provide network reachability information between peers. BGP route servers [3], [4] are frequently deployed by IXP operators to help interconnect BGP peering for members.

By using Software Defined Networking (SDN) with OpenFlow [5] and a programmable data plane language such as P4 [6], the operator has fine-grained control on how traffic flows within their network. In contrast to the more limited nature of the configuration interface provided by legacy vendor switches. For this reason, non-programmable switches drastically limit the possibilities and innovation of the IXP management plane.

IXPs enforces strict rules [7], [8] to help restrict the effects of the layer-2 broadcast, unknown-unicast and multicast traffic. One of the most common approaches is that a member must provide the MAC address of their connecting router in advance to the IXP. The IXP will in turn use it to configure a MAC filtering Access Control List (ACL) on the switch port designated for the member to connect to [9].

To further limit the potential for bad traffic entering the IXP, IXP operators typically place new connections from members in a quarantine VLAN [10]. Once it is clear that a member's connection is not transmitting any bad traffic, the quarantine VLAN is removed and the connection is migrated over to the peering VLAN.

The Umbrella [11] architecture is an evolution of this philosophy and utilizes SDN to bring this down to the switching fabric. The Umbrella architecture achieves this by translating broadcast traffic into unicast traffic, which prevents the IXP participants from receiving undesired traffic. Additionally, it strictly enforces the allowed ethertypes as designated in [7], and drops any unknown/bad traffic, including those from connections not configured.

HolistIX is a further evolution on this, providing IXPs a platform that *a)* enforces best practices through a zero-touch provisioning interface and *b)*, reduces the complexity required to run an IXP. An important aspect of zero-touch is to help increase automation into network operations, enabling faster, more secure and less error-prone network deployments. Automation enables configuring networks based on a top-down approach, and looking at networks holistically when generating configurations.

We have identified that the push-on-green [12] approach, most commonly used within site reliability engineering (SRE), can be adapted and applied to automation within IXP networks. Push-on-green is commonly summarized in the following way: if all the tests come back as green (they all pass), the configuration is good, and therefore it can be pushed to production. For HolistIX we use it in the following ways:

1) Any network configurations need to be tested before they are pushed into the production network.
2) The operator is the ultimate arbiter when deploying. The network configuration generation and testing is

978-3-903176-36-2 ©2021 IFIP

automated, but ultimately the operator can review the test results and the configuration itself before pushing.
3) Previous deployment configurations are kept, so that the network can be rolled back to a previous working state automatically if issues arise during the deployment itself, or detected after the fact.

In section two, we introduce our architecture in detail. In section three, we go through the workflow underneath. In the last section, we briefly discuss our experience deploying and using our solution in two IXP in production with tens of members each.

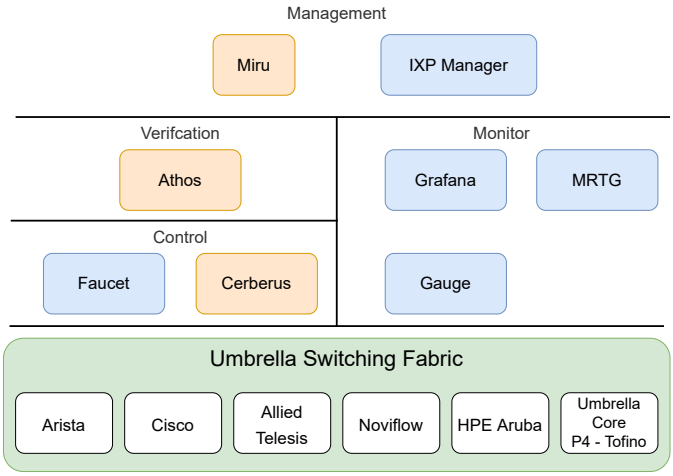

Fig. 1. HolistIX Full Stack Overview

II. HolistIX architecture

In this section, we introduce the architecture of HolistIX. Fig. 1 gives a full stack overview of the components and their general role within HolistIX. As indicated within the yellow boxes, the new components that have been designed for HolistIX are:

1) *Miru* — A diagramming module to plan the network.
2) *Athos* — Network verification module.
3) *Cerberus* — Network configuration generation and deployment module

A. IXP Manager

IXP Manager [14] is a widely used full-stack management platform within the IXP community. As of writing, there are currently 904 IXPs registered within PeeringDB [13] of which at least 176 [15] IXPs use IXP Manager. IXP Manager provides IXPs a platform that includes the ability to manage their members, their own infrastructure, and it both teaches and enforces best practices (i.e., prevent IP duplication). To help with running an IXP, IXP Manager has included support for popular third party packages that allows IXP operators to use existing tools for monitoring, looking glass, etc. There is also a customer portal included for members that allows for greater transparency between an IXP, and its members. IXP Manager greatly helps lower the barrier of entry for new IXPs to start, and existing IXPs to better manage their networks.

However, while IXP Manager greatly helps with the administration management and monitoring of an IXP, it does not provide any tools for managing the switching fabric itself. By adding HolistIX as zero-touch network configuration, it allows IXP Manager to be a truly full-stack management platform, from web down to switching fabric.

B. Miru

The first component of HolistIX is the planning and diagramming module, called Miru. By exploiting the information already available within IXP Manager (i.e., switch and member connections), Miru allows IXP operators to plan their network topology by diagramming them. Traditionally, the diagramming of the network gets left until the end of a deployment process, and it is most often the first aspect that gets neglected.

With HolistIX, diagramming is brought to the forefront and is used to specify the network topology and acts as the source of truth for the network state. This is achieved through a simple drag and drop interface that is built on-top of the open source backend of diagrams.net [16] (previously known as draw.io).

The operator simply drag switch objects onto the canvas and then drag links between switches to connect them. Each switch object contains all the switch information (i.e., name, IP address, switch ports, etc.) as well as the member details (i.e., switch port connected to, peering VLAN member belongs to, IP and MAC addresses) connected to it. When drawing links between switches, Miru enforces a best practice philosophy by finding available core ports on each switch and associating them with the link that's been created. Miru allows the ports to be changed, but only to another core port. For any changes involving members, for example a new member connecting, Miru will update the switch information based on the information provided within IXP Manager. As the infrastructure remains unchanged, the topology remains unchanged as well.

After the operator finished diagramming the topology to reflect that of the physical topology running within the production infrastructure, the operator starts the verification process through a single click. Miru translates the diagrammed topology into a JSON representation. The topology JSON then acts as the source of truth for the network.

Fig. 2 shows an example topology JSON that has been translated from the finished topology diagram in fig. 3. The completed diagram shows a classic spine and leaf topology, here the *s1* and *s2* are OpenFlow edge switches and, *c101* and *c102* are P4 enabled core switches.

Once generated, the JSON is transferred to Cerberus, which generates a candidate network configuration. Athos will take topology JSON as input and builds an emulated network that is used to verify the candidate network configuration.

Miru has access to real time reports of the verification process that the operator can review as the tests run. Once the operator is satisfied with the results from the tests, they can deploy the candidate configuration to the production network.

C. Athos

To help IXP operators verify the network configurations that are generated for the IXP, we designed a push-on-green module that HolistIX uses, called Athos. Having a push-on-green module allows operators to automate the testing of the generated configuration and verify the state of their infrastructure network in a secure and controlled manner. As HolistIX, and IXPs, primarily focus on layer-2 connectivity between its members, Athos primarily tests for reachability and connectivity between members within the IXP. It does not test for BGP peering between members.

Athos takes the JSON representation of the topology generated by Miru, and proceeds to build an emulated network by using a modified Mininet [17]. The main modifications made to Mininet were to add IPv6 support, VLAN tagging for hosts and incorporate bmv2 for P4 software switches.

Athos configures the switches as either an OpenFlow Switch, or a bmv2 [18] switch that's been compiled with P4 Umbrella [26] code. Once Athos has configured the switches, it will then proceed to configure the hosts to represent the member connections that come into the IXP. Each host will be configured with the same IPv4/6 and MAC addresses as specified in IXP Manager, and will be VLAN tagged as designated.

After completing the network configuration, Athos will either start the Faucet [19] SDN controller or use a local instance of Cerberus to configure the switches. At this stage, Athos initiates a simple reachability test. This test goes through each of the hosts and checks if they are able to ping all the hosts within the same peering VLAN. The reachability tests check connectivity on both IPv4 and IPv6, depending on the member's configuration.

```
{"switch_matrix": {
    "dp_ids": { "s1": 1, "s2": 2},
    "links": [
        ["s1",   "11", "c101", "11"],
        ["s1",   "24", "c102", "14"],
        ["s2",   "12", "c101", "12"],
        ["s2",   "22", "c102", "22"],
        ["c101", "18", "c102", "18"],],
      "p4": ["c101", "c102"]},
  "hosts_matrix": [{
    "name": "h1",
      "interfaces": [{
        "switch": "s1",
        "switch_port": 1,
        "mac": "00:00:00:00:00:01",
        "ipv4": "10.0.0.1/24",
        "vlan": 100,
        "tagged": true }]}, {
    "name": "h2",
      "interfaces": [{
        "switch": "s2",
        "switch_port": 1,
        "mac": "00:00:00:00:00:02",
        "ipv4": "10.0.0.2/24",
        "ipv6": "fd00::2/64",
        "vlan": 100,
        "tagged": false }]}
]}
```

Fig. 2. Example of the topology translated to JSON

The next step, after finishing the reachability steps, is ensuring that the redundancy groups are functioning as intended. Athos checks if a source switch has more than one path to a destination switch, and shuts down the primary link if a backup path exists. Athos repeats the reachability tests, as described above, to ensure backup paths are now active. Once completed, Athos restores connectivity on the main link and repeats the reachability tests. Athos proceeds to repeat this process for all links that have redundancy enabled. Any links where redundancy testing is not possible (i.e, a switch with only one core link) does not get turned off, and Athos reports these back to Miru.

Once completed, Athos reports a summary back to the operator with the results of the verification process and whether the candidate configuration as well as the topology is valid.

While it is possible to run Athos by itself, as shown in Fig. 4 it is designed to run within a docker instance as part of HolistIX. In doing so, the testing network and the production network are separate.

D. Cerberus

Cerberus is the component that generates the network configuration as well as the deployment component, and can be considered as the controller of HolistIX. However, some of the core differences it has to a traditional SDN controller is that *1)* it has support for generating configurations for itself, as well as the Faucet [19] SDN controller, and *2)* it proactively configures the OpenFlow switches as opposed to doing learning based on the traffic flowing through the network.

1) Network Configuration Generation: Due to the neutral nature of IXPs, members generally prefer that their traffic remains untouched. HolistIX embraces this by focussing purely on layer-2 forwarding. IXPs are also more stable with very little changes happening inside the infrastructure itself, and member changes typically need to be preapproved by the IXP itself.

The configuration that Cerberus generates takes this philosophy down to the switching fabric. Based on the topology JSON provided via Miru, Cerberus uses the location and addresses of the members and proactively generate OpenFlow rules to enable interconnectivity between them.

Any traffic that comes into the network will go through our finite state machine in Fig. 5, where multicast traffic will be transformed into unicast traffic and sent to the destination member. For destination members not directly connected to the source switch, the destination MAC address gets rewritten based on the Umbrella concept

Cerberus also incorporates redundancy awareness when determining the route to the destination member. Based on Dijkstra's Shortest Path First algorithm, Cerberus computes redundancy paths, with backup paths recomputed by removing the primary link, until there are no redundancy paths left.

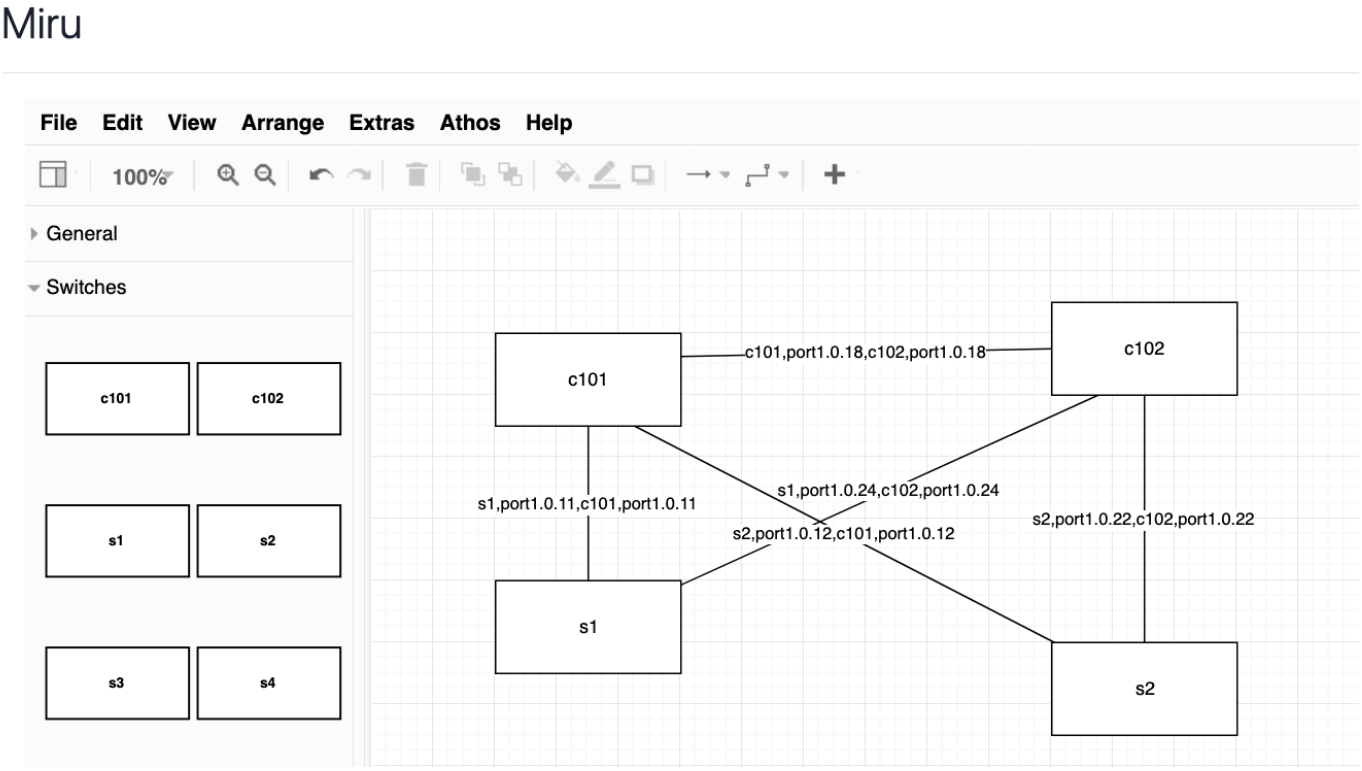

Fig. 3. Example of Miru interface

Fig. 6 shows an example of a simple topology, with each link having the same weight and the redundancy paths chosen based on the shortest path, with path 1 being the primary path, followed by path 2 as the primary backup, if both those fail, it resolves to path 3. Every switch contains a redundancy group to all the other switches that have members connected to them.

2) Pushing to production: IXPs move large volumes of traffic through them, and members expect to be connected to a stable interconnect.

One of the primary goals of Cerberus' deployment modules is to have as little as possible of an impact on the running network. When deploying a new configuration, Cerberus pulls and compares the state of the running network with the new verified configuration. Cerberus then determines the minimum amount of changes required to update the configuration.

The primary function for Cerberus is to ensure that the state within the switching fabric matches what is configured in IXP Manager, diagrammed in Miru and verified through Athos. To help verify the state of the running network, Cerberus will periodically pull and compare the running state and ensure that it matches the configured state.

Cerberus also has a rollback function that allows IXP operators to restore the network to a previously working state. The rollback also allows the network to roll back to its previous working state when it detects errors during the deployment process (i.e., a switch is unreachable). On top of returning the network back to the previous state, Cerberus will also retain a copy of the failed configuration. This allows operators to quickly get their network back up and running, as well as get a better understanding as to why a previous deployment failed.

Additionally, when configuring switches, Cerberus also stores a local copy of the configuration, allowing the switch to configure itself to the last known stable configuration on boot up. This allows the network to be up and running as quick as possible, since the switch does not have to wait for the controller to configure it.

III. Workflow

The vision for IXP Manager is to simplify the creation of IXPs. HolistIX builds on this vision by automating the network configuration within an IXP. This is achieved by allowing the IXP operator to completely configure their network without leaving IXP Manager.

Fig. 7 shows a sample of the typical workflow that is carried out within HolistIX after physical cabling has been completed. The operator actions and what is observed are declared with the solid boxes, and the automation elements with dashed lines. This section will go over the two primary workflows that operators will go through when using HolistIX.

A. Topology declaration

In order for HolistIX to configure the network, it needs to be aware of how the physical topology looks. IXP Manager allows operators to add switches via SNMPv2, through which it will obtain core switch details (i.e., hostname, make, model, address, etc.), port states, optic availability, etc.

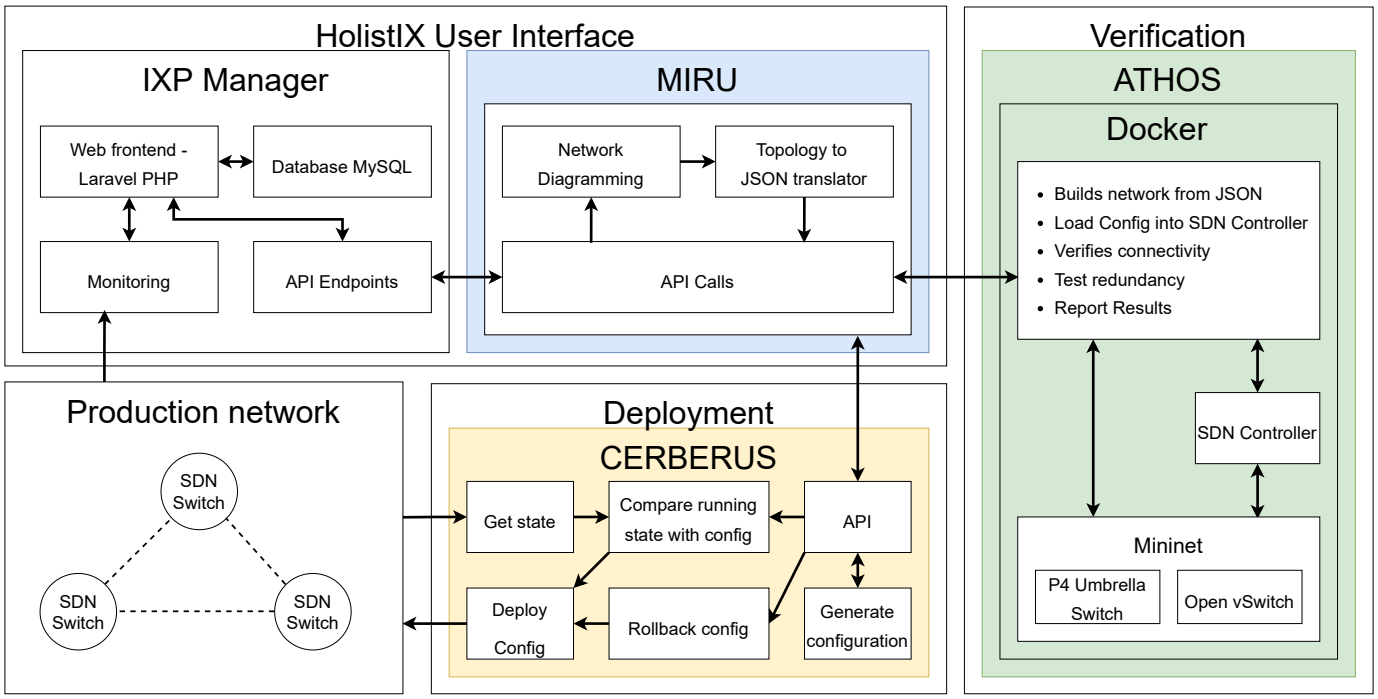

Fig. 4. HolistIX Architecture

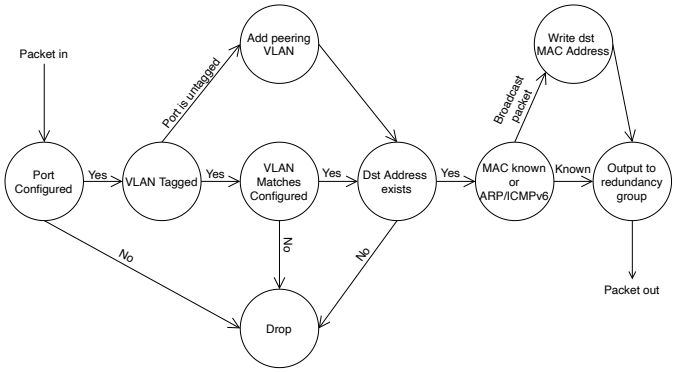

Fig. 5. Finite state machine of traffic flowing through a switch

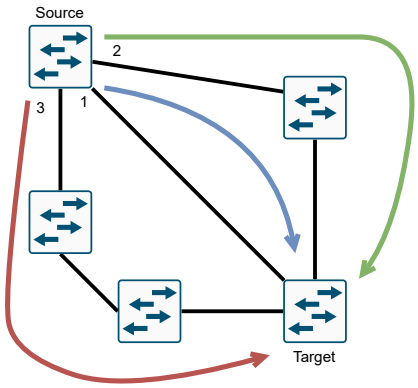

Fig. 6. Illustration on how redundancy paths are determined

With the switches configured, the operator now needs to add members and let IXP Manager know where they are connected into their network. This is also where the peering VLAN, IPv4/6 and MAC addresses is specified.

Miru's integration into IXP Manager allows the operator to remain within IXP Manager and navigate to Miru via the sidebar. On load, Miru will grab all the switch and member details from IXP Manager and populate the Miru sidebar with switch objects as seen in Fig. 3. The operator drag the switch objects onto the canvas, and declares internal links by hovering over a switch and dragging the connection arrow from the source switch to the destination switch. When connecting the switches, each connection receives a label based on the declaration, as seen in Fig. 3

After declaring the topology, the operator starts the testing and verification process by clicking on the "Athos" menu item. In the background, the generation of the network configuration runs, followed by the verification process in Athos. Miru will continuously report to the operator the results from Athos as they are running, allowing them to identify issues as they occur. Upon completion of the verification process, the operator can review the results, as well as the configuration itself. When satisfied, they click on the "Deploy" button, which deploys the configuration to the running network.

Thus configuring the network without the operator leaving HolistIX and without touching the CLI on the switches.

B. Changing member details

The process of adding, removing or updating members can also be performed without the operator leaving HolistIX.

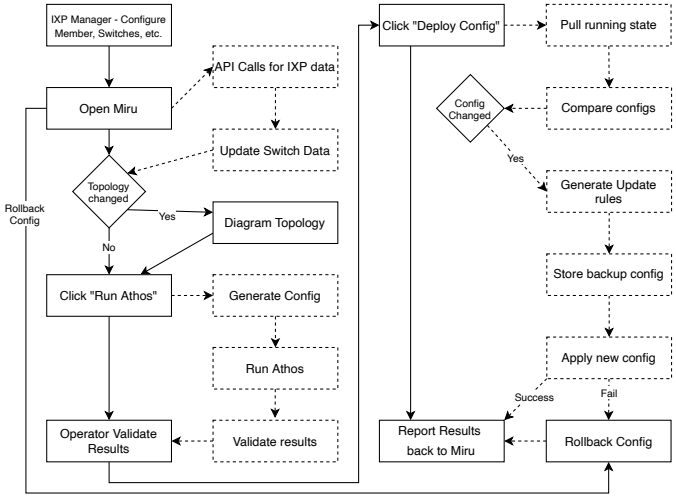

Fig. 7. Workflow of HolistIX

The IXP operator makes the appropriate changes within IXP Manager (i.e., Add Member, change IP, etc.), and when they navigate to Miru the switch objects will be automatically updated to reflect the changes made in IXP Manager.

If the topology remains unchanged, they do not need to redraw the topology, but can simply run Athos and finally deploy. By having a proactive design, members can be safely connected to the network before they are configured to receive traffic. Traffic will be dropped at the edge of the network until it has been configured through HolistIX. The inverse also applies, as connected members configured as "disconnected" will lose connectivity into the IXP before they are physically disconnected.

IV. Deployment at IXPs

A. ToulX

Toulouse Internet eXchange (TouIX) is a non-profit and neutral IXP founded in 2005 in Toulouse, France. In collaboration with LAAS-CNRS, transitioned to an Openflow SDN based architecture in 2015 called Toulouse SDN Internet eXchange (TouSIX) [20], [21]. In doing so, TouIX became the first SDN based IXP within Europe.

TouIX consists of three Point of Presences (PoP)s with one OpenFlow each. Each PoP acts as an edge switch, configured in a triangle topology, and therefore does not require a core switch. However, due to the hardware limitation of the previous switches, TouIX did not support IPv6.

In April 2020, TouIX was migrated over to a beta version of HolistIX, retaining the same topology but now with Allied Telesis x930 switches at each PoP. Allowing TouIX to now fully support IPv6. The beta version of HolistIX consists of Cerberus generating a config for a Faucet Controller by utilizing the IXP Manager API. This deployment did not have support for Miru or Athos, so any topology changes had to be configured within Cerberus. Due to how rare infrastructure and topology changes are within an IXP, the topology configuration remains mostly unchanged. The primary focus for Cerberus in this deployment was the zero-touch approach to updating members on the switching plane, as they are updated within IXP Manager. Due to this, the config generated for TouIX only works with topologies configured in a triangle.

All participants have been migrated to the new Allied Telesis switches with minimal disruptions.

B. DIX-IE

The Distributed IX in Edo(DIX-IE) is an IX based in Tokyo, Japan, that is used for experiments by the WIDE project. Formerly known as NSPIXP-2, DIX-IE is a historical IX that has existed since 1995. In 2014, the University of Tokyo started the Programmable IX in Edo(PIX-IE) [22], to explore the possibilities of incorporating SDN in an IXP.

While both the PIX-IE and the TouSIX project started at around the same time, initially they did not share any relationship with one another. At the end of 2016, the teams from both projects started to collaborate, extensively sharing their knowledge and experiences.

With the knowledge gained during the HolistIX deployment in April 2020, the scope of HolistIX was expanded. The results of this are the addition of *a)* Athos, for added security in deployments by testing and validating the configurations that are generated, *b)* Miru, a centralized location for diagramming the network topology and interacting with the different components of HolistIX and *c)* Cerberus' native Openflow deployment module. The updated Cerberus allows for dynamic topology changes, and it is no longer restricted to a fixed triangle topology.

With HolistIX, the migration from DIX-IE to PIX-IE aims to provide added value to the IXP by making use of intent based networking and taking a zero-touch approach to managing an IX. The migration from the old DIX-IE to the new PIX-IE will be carried out in the second half of 2021.

C. Operational experience

Both migrations brought us difficulties similar to any other technical migration. What we observe as the tangible contribution is the zero-touch and monitoring brought from the HolistIX concept. The time and risk to add new members to the IXP or modify one was made with no time and no risk. We have not observed any interruption of service.

V. Discussion

The core function of an IXP is to act as an interconnect between multiple peer participants, where member changes are kept to as minimal as possible. This is different to cloud provider networks that have much larger volume of changes, upwards of thousands a month. Each of these changes need to keep QoS, reliability and affected services in mind. Due to this, a lot of current research for zero-touch networks involve using AI [23], [24] or deep learning [25]. IXPs on the other hand are large layer-2 networks that are neutral in the traffic that it passes between its members. Therefore, IXPs are a lot simpler compared to cloud and campus networks, and introducing

AI would add unnecessary complexity to the network and deploy process. Keeping in line with reducing complexity within the network, the configuration that HolistIX generates also consists of purely layer-2 forwarding. This is especially true when looking at the P4 switches that is configured to do forwarding based on the Umbrella concept. A benefit of this approach is that an IXP can migrate to using HolistIX with no requirements from its members nor impact on their traffic.

Since the focus is on layer-2 forwarding, the connectivity tests within Athos are all done as ping tests. By keeping to this philosophy, it is easier for operators to spot any potential issues before they make it to the production network.

The approach that HolistIX takes to zero-touch network management is to introduce automation and validation right through the pipeline. However, requiring operators approval for deployment was a deliberate decision made to keep errors deployed into production to a minimal.

VI. Future Work

HolistIX is primarily focused on working with the stability of an IXP, where internal infrastructure changes are typically infrequent, and member changes are known in advance. It is, however, not limited to IXPs. As Miru uses API calls to retrieve host and member details, it can be easily adapted to work with other management platforms, without any changes required to Athos and Cerberus. This opens HolistIX to be used in other environments where the nodes within the network is known and pre-provisioned like in datacenter networks.

VII. Conclusion

In this paper, we introduced a zero-touch full-stack architecture called HolistIX. The "push-on-green" has inspired our design choices, resulting in creating a workflow in three main steps: declaring, verify/testing, and push in production with roll-back capacities at any given point. The names Miru, Athos, and Cerberus are for the three blocks that, respectively: diagram to declare, verify and test with, and deploy in production.

Our deployment at the TouIX and DIX-IE to PIX-IE projects at real IXPs shows that our approach has sufficient maturity to be deployed and operated. We have demonstrated the architectural benefits to the two IXPs bringing to both a high-level of simplification and robustness. Furthermore, a more stable and flexible switching fabric offers additional benefits such as increased security and services for participants

From the top of the stack, with a full drag and drop interface, to the bottom, with a OpenFlow and P4 coded data plane, HolistIX maximizes flexibility and openness. HolistIX as a zero-touch provisioning solution is the first iteration towards a more autonomous and API-driven solution not only for IXPs but for high-performance transport programmable solutions.

Acknowledgment

The authors would like to thank all WIDE project participants and founders and all the Toulouse IXP members.

References

[1] B. Ager, N. Chatzis, A. Feldmann, N. Sarrar, S. Uhlig, and W. Willinger, "Anatomy of a large european IXP," ACM SIGCOMM Computer Communication Review, vol. 42, no. 4, p. 163, 2012.

[2] Y. Rekhter, S. Hares, and D. T. Li, "A Border Gateway Protocol 4 (BGP-4)," RFC 4271, Jan. 2006.

[3] P. Richter, G. Smaragdakis, A. Feldmann, N. Chatzis, J. Boettger, and W. Willinger, "Peering at Peerings," Proceedings of the 2014 Conference on Internet Measurement Conference - IMC '14, pp. 31–44, 2014.

[4] E. Jasinska, N. Hilliard, R. Raszuk, and N. Bakker, "Internet Exchange BGP Route Server," RFC 7947, Sep. 2016.

[5] Open Networking Foundation, "OpenFlow Switch Specification Version 1.3.1", https://opennetworking.org/wp-content/uploads/2013/04/openflow-spec-v1.3.1.pdf — Accessed 12-08-2021.

[6] Pat Bosshart, Dan Daly, Glen Gibb, Martin Izzard, Nick McKeown, Jennifer Rexford, Cole Schlesinger, Dan Talayco, Amin Vahdat, George Varghese, and David Walker. 2014. "P4: programming protocol-independent packet processors." *SIGCOMM Comput. Commun. Rev. 44*, 3 (July 2014), 87–95.

[7] AMS-IX, "Allowed Traffic Types,", https://www.ams-ix.net/ams/documentation/allowed-traffic — Accessed 14-08-2021

[8] M. Hughes, M. Pels, and H. Michl, "Internet Exchange Point Wishlist", https://www.euro-ix.net/en/forixps/ixp-wishlist/ — Accessed 14-08-2020

[9] Open-IX, "Internet Exchange Point Technical Standards", https://www.open-ix.org/en/standards/ixp-technical-standards/ — Accessed 14-08-2020

[10] Andy Davidson, "Building an IXP", https://www.menog.org/presentations/menog-6-7-8-9/Building_an_IXP.pdf — Accessed 14-08-2020

[11] M. Bruyere, et al., "'Rethinking IXPs' architecture in the age of SDN," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 12, pp. 2667-2674, 2018.

[12] Daniel V. Klein, Dina M. Betser and Mathew G. Monroe, "Making 'Push On Green' a Reality: Issues & Actions Involved in Maintaining a Production Service." *login*, vol. 39, no. 5, pp 26-32, 2014

[13] PeeringDB, PeeringDB https://www.peeringdb.com/ — Accessed 02-08-2021.

[14] IXP Manager, "IXP Manager — About IXP Manager", https://www.ixpmanager.org/about — Accessed 22-08-2021

[15] IXP Manager, "IXP Manager — World's most trusted IXP platform", https://www.ixpmanager.org/ — Accessed 02-08-2021

[16] Diagrams.net, https://www.diagrams.net/ — Accessed 02-08-2021.

[17] Mininet, "Mininet: An Instant Virtual Network on your Laptop (or other PC)", http://mininet.org/ — Accessed 02-08-2021

[18] Barefoot Networks, "Behavioral Model version 2", http://bmv2.org/ — Accessed 02-08-2021.

[19] Faucet, "Faucet SDN Controller", https://faucet.nz/ — Accessed 02-08-2021.

[20] Toulouse Internet eXchange, "TouSIX Project", http://www.touix.net/en/content/tousix-project — Accessed 10-08-2021.

[21] R. Lapeyrade, M. Bruyère and P. Owezarski, "OpenFlow-based migration and management of the TouIX IXP," NOMS 2016–2016 IEEE/IFIP Network Operations and Management Symposium, 2016, pp. 1131-1136

[22] PIX-IE, https://pix-ie.net/index-e.html — Accessed 10-08-2021.

[23] C. Benzaid and T. Taleb, "AI-Driven Zero Touch Network and Service Management in 5G and Beyond: Challenges and Research Directions," in *IEEE Network*, vol. 34, no. 2, pp. 186-194, March/April 2020

[24] G. Carrozzo et al., "AI-driven Zero-touch Operations, Security and Trust in Multi-operator 5G Networks: a Conceptual Architecture," *2020 European Conference on Networks and Communications (EuCNC)*, 2020, pp. 254-258

[25] D. Bega, M. Gramaglia, M. Fiore, A. Banchs and X. Costa-Perez, "AZTEC: Anticipatory Capacity Allocation for Zero-Touch Network Slicing," IEEE INFOCOM 2020 — *IEEE Conference on Computer Communications*, 2020, pp. 794-803

[26] Marc Bruyere, Remy Lapeyrade, Eder Leão Fernandes, Ignacio Castro, Steve Uhlig, Andrew W. Moore, Gianni Antichi, "Umbrella: a deployable SDN-enabled IXP Switching Fabric" *SOSR* 2018