ENDEAVOUR: Towards a flexible software-defined network ecosystem

ENDEAVOUR

Project name	ENDEAVOUR
Project ID	H2020-ICT-2014-1 Project No. 644960
Working Package Number	4
Deliverable Number	D4.4
Document title	Implementation of the Selected Use Cases for IXP Operators
Document version	0.8
Editor in Chief	Kopp, DE-CIX
Authors	Kopp, Dietzel, Abt, Chiesa, Fernandes
Date	31/01/2017
Reviewer	Chiesa, UCL
Date of Review	01/01/2017
Status	*Public*

Revision History

Date	Version	Description	Author
15/11/16	0.1	First Skeleton	Kopp
29/11/16	0.2	Description and Demo of Broadcast Prevention	Fernandes
30/11/16	0.3	Description of Load Balancing	Kopp
05/12/16	0.4	Minor changes	Kopp
06/12/16	0.5	Description of Access Control	Chiesa
08/12/16	0.6	Internal review	Canini, Chiesa
13/12/16	0.7	Review incorporated	Kopp
13/12/16	0.8	Final version	Kopp

Executive Summary

Over the course of the ENDEAVOUR project the consortium developed a wide range of use cases as potential candidates to be implemented within the ENDEAVOUR prototype. After consolidating the most compelling use cases we implemented them into the ENDEAVOUR platform. To this end, the present deliverable showcases the implemented use cases of the ENDEAVOUR platform for Internet eXchange Point (IXP) operators. Each use case is demonstrated in a video. In addition, Deliverable 4.5 discusses the relevant use cases for IXP members. In combination, these two deliverables reflect the current state of the ENDEAVOUR platform prototype. We present technical background necessary to understand the implementation of each use case, the high level implementation itself, as well as a workflow of each demonstration.

Contents

1 Introduction

In this report, we present the implemented ENDEAVOUR use cases for IXP operators. This includes demo videos for all use cases and a description of the implementation of all demonstrators. We first outline the setup of the development environment in Section 2. We then present the implemented use cases (Section 3) together with technical details. Next, we discuss the workflow of the demonstrators (Section 4). The final Section 5 summarizes the report.

Note that this document is not self-contained but it accompanies the demonstrator videos[1] and the code[2] available in the ENDEAVOUR GitHub repository. Details on the implementation of the ENDEAVOUR Software Defined Networking (SDN) and monitoring architectures can be found in Deliverable 2.3 and 3.3, respectively. For a comprehensive description of the use cases for IXP operators we refer to Deliverable 4.2.

2 Development Environment

The ENDEAVOUR consortium maintains its code on the GitHub platform. We agreed on only having deployable code in the master branch. The development is coordinated by weekly calls and the ENDEAVOUR Waffle board[3]. Additionally, the ENDEAVOUR GitHub repository provides up-to-date setup instructions to provision a Virtual Machine (VM) with the ENDEAVOUR platform and install the necessary software for running the ENDEAVOUR platform.

Before the use case specific deployment scripts can be started, the ENDEAVOUR platform must be installed. The following commands set up the environment in a vagrant VM:

```
$ git clone https://github.com/h2020-endeavour/iSDX.git
$ cd ~/iSDX
$ vagrant up
$ vagrant ssh
$ git clone https://github.com/h2020-endeavour/endeavour.git
$ cd ~/iSDX/test
$ sh buildall.sh
```

[1]https://www.youtube.com/playlist?list=PL16z513p1ClFZqcITtxGTbgCe0wQOU0Ew
[2]GitHub - https://github.com/h2020-endeavour/endeavour
[3]Waffle IO - https://waffle.io/h2020-endeavour/endeavour

As a result the current version of the platform is ready to run specific use cases or tests for use cases. A use case test can be started with the following command, "test-name" refers to a specific test:

```
$ sudo bash startup.sh --stats test-name
```

Further details on the platform and the test architecture can be found in Deliverable 2.3.

3 Implemented Use Cases for Operators

In this section, we present the selected use cases for IXP operators, namely, Broadcast Prevention, Access Control, and Load Balancing. For each use case, we briefly discuss the motivation, a high-level description of the implementation, and how we evaluated the correctness of the code. Before, we outline the reasons for the selection of these three use cases.

3.1 Selection Process

During the ENDEAVOUR project a large number of different use cases have been collected, discussed, and analyzed. An extensive collection of use cases from related work can be found in Deliverable 4.1. These use cases, in addition to those developed by the ENDEAVOUR consortium and the ones discussed at the IXP member workshop (summary in Deliverable 5.3) were joined and structured. With this comprehensive list at hand we were able to present our most promising candidates to a wider audience during the RIPE71 meeting. As a result we compiled a list with an extensive motivation, summary of the current situation, and a technical description, documented in Deliverable 4.2.

Eventually, DE-CIX's operational insights combined with even more feedback of the customers helped to select the first set of promising candidates to be implemented. The positive Extended Advisory Board feedback workshop (summary in Deliverable 5.5) encouraged us to proceed with the implementation of the selected set.

3.2 Implementation of Use Cases

Following, we discuss the implementation of the three selected use cases for IXP operators and give a brief motivation and evaluation.

3.2.1 Broadcast Prevention

Broadcast traffic from the Address Resolution Protocol (ARP) enables the discovery of Media Access Control (MAC) addresses at Internet eXchange Points (IXPs). ARP requests are typically flooded in the network when a member's router needs to forward traffic through another network device but it still does not know the MAC address of that intermediate device. While discovering MAC addresses is a fundamental operation for correctly running an IP network, the excess of broadcast ARP packets wastes bandwidth and can be harmful for a large domain such as a large IXP network. ARP storms can cause disruptions and even bring a whole network down.

The mitigation of broadcast at an IXP is one of the goals of the Umbrella component. From the fact that MAC addresses of the peering routers and the ports connected to the IXP fabric are known in advance to the IXP operators, the Umbrella component can preemptively compute well-crafted MAC addresses for each of its network device so that the path to reach each device is written within the MAC address. With SDN capabilities this process can easily be controlled by the network operator. In the ENDEAVOUR platform every broadcast ARP is turned into an unicast packet by flows installed in the OpenFlow switches of the fabric. We refer the reader to the Deliverable 2.2 for a comprehensive description of the Umbrella component and how the MAC encoding prevents broadcast ARPs. Moreover, the Deliverable 2.2 also contains a detailed evaluation of Umbrella including a comparison with traditional methods to prevent broadcast storms such as ARP sponge.

3.2.2 Access Control

Controlling what traffic is allowed to traverse a network is a crucial, yet cumbersome, operation in today's IXP networks. With hundreds of members sharing the same physical infrastructure, IXP operators must carefully configure their network devices so as to curb any possible source of malicious or unwanted traffic. Yet, traditional IP networks do not have the necessary fine-grained capabilities required to prevent, for instance, BGP sessions external to the IXP fabric to be established throughout the IXP network (a bad practice to be avoided). SDN has the potential to increase the level of security and safety. It allows to further limit the allowed traffic exchanged via an IXP network, while it filters out packets due to misconfiguration of a member's router.

The implementation of access control capabilities is part of the Access Control module, which can be found in the `endeavour/acctrl` folder of the ENDEAVOUR main repository. The structure of the Access Control module is built using the same ideas described in Deliverable 3.3 (Section 3.1) for the monitoring module. The access-control rules are installed in a dedicated table that is located right after the Main-Out table. A controller receives access-control rules to be installed in the IXP platform. Those rules are formatted in JSON schema and can easily implement both white- and black-list filtering policies. An example of such rules can be found in `iSDX/test/specs/test1-mh-ac-access_control_flows.cfg`. We refer the reader to Deliverable 2.3 (Section 3.9) for more details on how the access control rules are structured at the forwarding plane level.

As with all other selected use cases, we evaluated the Access Control module in a Mininet[4] environment. We refer the reader to Section 4.2 of this deliverable to get a sense of the expressiveness achievable with fine-grained access-control policies.

3.2.3 Load Balancing

Load balancing traffic across multiple network paths is a crucial operation for any sufficiently large IXP infrastructure. It entails the steering of traffic flows so as to optimize network performance. In an IXP topology, which is often a core-edge symmetric network, the idea is to balance traffic among the multiple available paths that exists between any two members through a core switch. Load-balancing traffic is known to both reduce congestion and improve resiliency to network failures as balancing traffic can help to easily maintaining network connectivity. As already described in greater detail in Deliverable 4.2, traditional routing already provides mechanisms, e.g. ECMP, to load balance traffic. Nevertheless, traditional implementations lack in flexibility on how the traffic can be weighted between the available paths. We believe SDN provides the greater flexibility and control needed to support load balancing mechanisms at IXPs.

Next, we describe the current state of the software implementation within ENDEAVOUR and show a preliminary evaluation. Finally, we give a demonstration of the implemented features in Section 5.3.

The load balancing implementation of ENDEAVOUR leverages a layer 2 label switching mechanism, which is extensively described within deliverable Deliverable 4.2. Basic layer 2 label switching functionality is provided by the Umbrella component. On top of this ENDEAVOUR builds load balancing capabilities for the IXP infrastructure. The implementation of the

[4]Mininet - `http://mininet.org`

load balancing functionality is located within the Umbrella controller in ENDEAVOUR and can be found in `endeavour/uctrl/load_balancer.py`. It is structured in a modular way to enable for any later enhancement e.g. by more advanced balancing algorithms. A first method has been implemented for this use case, which leverages the IP address to distribute traffic across the infrastructure. In greater detail, the currently implemented method works by using the value of the last bit of source and destination IP address of the forwarded packets. These fields are used in combination as an identifier for the core switch to be used to send traffic traversing the IXP network. Finally, the necessary forwarding rules to steer traffic according to this load balancing schema are generated and installed onto the edge switches.

To evaluate the effectiveness of the Load Balancing function, several independent network streams are generated over the IXP infrastructure. By monitoring the bandwidth utilization of the switches, we can verify a correct distribution of the load amongst the IXPs core switches. This test scenario is part of the demonstration in section 4.

4 Demonstration of Use Cases

4.1 Broadcast Prevention

In this section, we show how ENDEAVOUR prevents broadcast ARP in the IXP fabric by encoding in the destination MAC the path a packet should follow in the network.

Demonstrator description. The Broadcast Prevention use case demonstrator is built upon the IXP topology shown in Figure 1. The IXP network is composed by four edge switches connected to four core switches. Eight members are connected to the edges as well as a Route Server and an Address Resolution Protocol (ARP) Proxy. Every member has a single host connected from different IP networks.

Two scenarios are used to show how Umbrella transforms broadcast traffic into unicast.

1. Hosts send traffic to the routers that have to forward it to the respective next hop peer.

   The demonstrator initially has all the members with a BGP session established with the Route Server and all the flows configured by the ENDEAVOUR platform are already installed in the OpenFlow switches

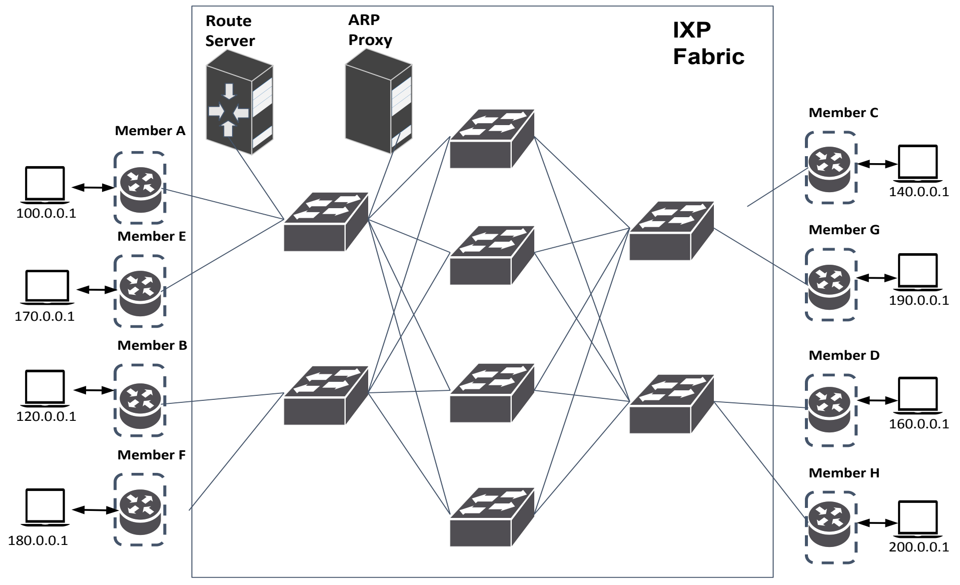


Figure 1: Topology for the Broadcast Prevention use case.

of the fabric. The following steps describe the sequence of events of the demonstrator:

1.1. Host `120.0.0.1` wants to send traffic to Host `200.0.0.1`. It generates a one minute flow of HyperText Transfer Protocol (HTTP) traffic to the destination.

1.2. Member `B` does not know the MAC address of member `H`, so it sends a broadcast ARP request to the fabric.

1.3. When the packet reaches the first switch the destination MAC of the ARP is encoded to the path of the ARP Proxy `01:07:00:00:00:00`. Each byte of the encoding shows the port a next hop should forward a packet. The packet is then sent to the next hop, a core switch.

1.4. The core switch matches the first byte of the MAC, since it has value `01`, the packet is forwarded through the port 1.

1.5. The packet reaches the edge connected to the ARP Proxy. The edge switch matches in the second byte of the destination MAC, rewrites the destination with the ARP Proxy MAC and then sends the packet through the port `07`.

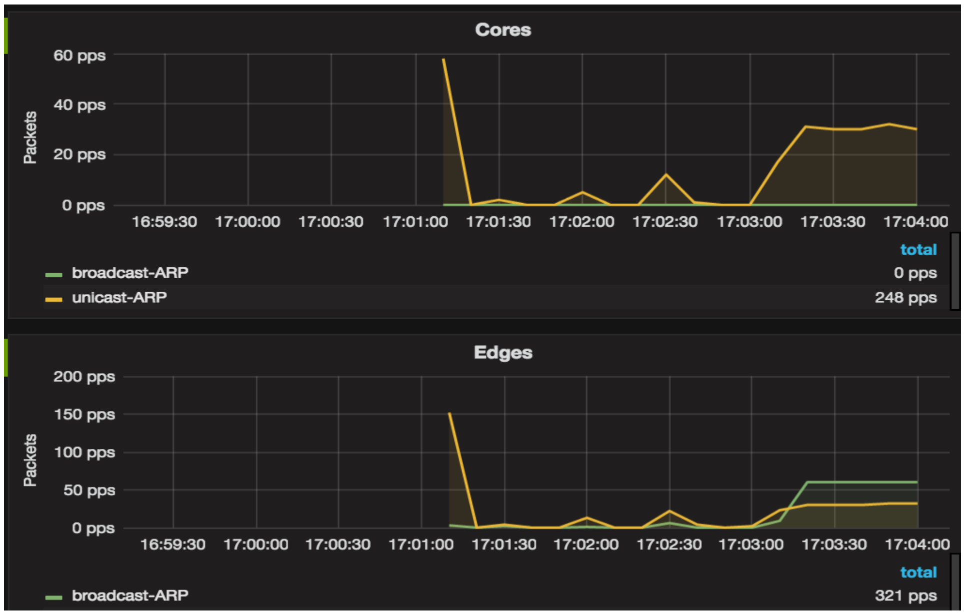


Figure 2: Resulting graphs for the Broadcast Prevention demonstration.

1.6. The ARP proxy receives the request and replies with member's `H` virtual MAC. The request goes through the same process of path encoding and reaches member `B`.

1.7. Member `B` now sends the traffic to member `H`.

1.8. In order to generate more ARP and traffic, more members send traffic to each other.

The resulting graph is shown in Figure 2, from time 17:01:00 to 17:02:30 where we show that no broadcast traffic reaches the core of the IXP.

2. Member's routers are disconnected from the IXP resulting in the loss of BGP connections between the route server and the respective members.

   Again, all members have a BGP session established with the Route Server and all the flows configured by the ENDEAVOUR platform are installed in the OpenFlow switches of the fabric. The following steps describe the sequence of events of the second scenario:

2.1. Six members are disconnected from the IXP fabric.

2.2. BGP sessions from the Route Server to these members are lost.

2.3. The ARP cache of the Route Server is flushed.
2.4. The Route Server will keep trying to establish the BGP session with the disconnected member. It will generate broadcast ARP requests in the fabric.
2.5. The flows installed by the Umbrella component will turn all the broadcasts into unicast packets.
2.6. Each packet have a distinct path encoded from the Route Server to their destinations. Therefore, no flooding occurs in the fabric.

Again, we can see in the Figure 2, from time 17:03:00 until the end that the total of broadcast packets in the core is zero.

Reproducing the use case demonstration. We used `torch` for orchestrating and scheduling the network events needed to showcase the Broadcast Prevention use case. The specification file containing each network event is stored in `iSDX/test/specs/test1-mh-arp.spec`. The monitoring rules needed to verify the correct behavior of the platform can be found in `iSDX/test/specs/ test1-mh-arp-monitor_flows.cfg`.

Grafana[5] is the tool used to reproduce the flow of traffic through the IXP fabric. The web-based Grafana interface can be accessed via any browser installed on the Host machine by entering the following address `http://localhost:3000`. The visual dashboard for the Broadcast Prevention use case can be imported from file `iSDX/test/arp_demo_dashboard.py`.

4.2 Access Control

In this Section, we highlight how the ENDEAVOUR platform takes advantage of SDN's direct control over packet-processing rules to enable members to express flexible fine-grained policies for access control.

Demonstrator description. The Access Control use case demonstrator is built upon the simplified IXP scenario depicted in Figure 3. The IXP operator wishes to (i) prevent BGP sessions involving non-border routers to be established throughout the IXP networks and (ii) filter out all the OSPF traffic that enters the IXP fabric.

The demonstration evolves as a sequence of three phases, each one spanning a 40 seconds time interval:

[5] Grafana Visualization Tool - `http://grafana.org/`

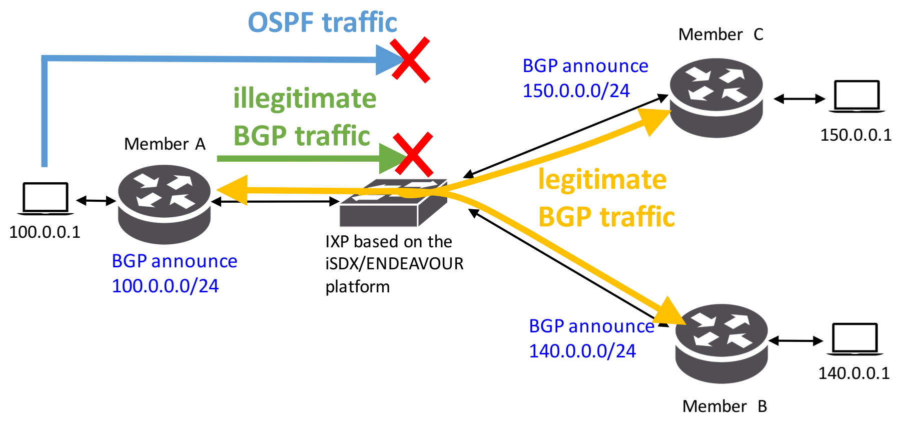


Figure 3: Simplified ENDEAVOUR SDX topology for the Access Control use case.

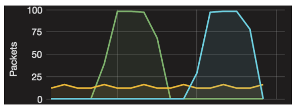
1. The network is started. BGP sessions among the three border routers and the Route Server are established and BGP traffic flows regularly throughout the IXP network.

2. Member A's border router generates BGP packets towards host 150.0.0.1, which a destination outside of the IXP fabric.

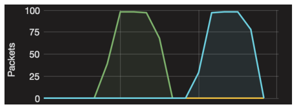
3. Host 100.0.0.1 generates OSPF packets towards host 140.0.0.1.

The above three phases are depicted in Figure 4a, where we use different colored lines to draw the different types of traffic traversing the IXP network: yellow for legitimate BGP traffic (Phase 1), green for illegitimate BGP traffic (Phase 2), and blue for OSPF traffic (Phase 3). Figure 4b and Figure 4c shows what traffic is being dropped and forwarded, respectively. We can observe that legitimate BGP traffic is being correctly forwarded while the illegitimate BGP and OSPF traffic is being dropped.

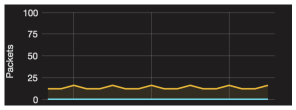
Reproducing the use case demonstration. We leveraged *torch* for orchestrating and scheduling the network events needed to showcase the Access Control use case. The specification file containing each network event is stored in `iSDX/test/specs/test1-mh-ac.spec`. The monitoring rules needed to verify the correct behavior of the platform can be found in `iSDX/test/specs/ test1-mh-ac-monitor_flows.cfg` and the access-control rules are in `iSDX/test/specs/ test1-mh-ac-access_control.cfg`. We leveraged Grafana to visualize the flow of traffic through the IXP fabric. The web-



(a) Generated traffic.



(b) Dropped traffic.



(c) Forwarded traffic.

Figure 4: Demonstration of the Access Control use case. Colors usage: yellow for legitimate BGP traffic, green for non-legitimate BGP traffic, and blue for OSPF traffic.

based Grafana interface can be accessed via any browser installed on the Host machine by entering the following address `http://localhost:3000`. The visual dashboard for the Access Control use case can be imported from file `iSDX/test/access_control_dashboard.py`.

A narrated video of this use case demonstration can be found at:

- `https://www.youtube.com/watch?v=8O_dZQWzyVU`

4.3 Load Balancing

In this section, we describe how ENDEAVOUR exploits the programmability of SDN to enable basic load balancing capabilities at an IXP network. We put forth a first simplified load balancing implementation that distributes traffic according to source and destination IP addresses of packets being forwarded.

Demonstrator description. The IXP setup is depicted in Figure 5. It consists of four core switches and three edge switches. Three IXP members are connected to the IXP platform, each to a different edge switch.

Member A is owns an Internet Protocol (IP) subnet `100.0.0.1/24` that includes four host machines. This variety of participants with networks and hosts actually provides a narrowed version of the actual situation at an IXP where Internet Service Providers (ISP) and Content Delivery Networks (CDN) provide connectivity to a wider range of user machines. When traffic starts to flow between the IXP members, the IXP aims to balance traffic within its core infrastructure so as to minimize packet dropping and reducing buffer latency at the forwarding devices. Any other behavior of traffic distribution can even lead to a severe situation, where specific links are overloaded and packet loss might occur.

To enable the Load Balancing use case within the ENDEAVOUR platform, we implemented a mechanism that performs a decision on the forward-

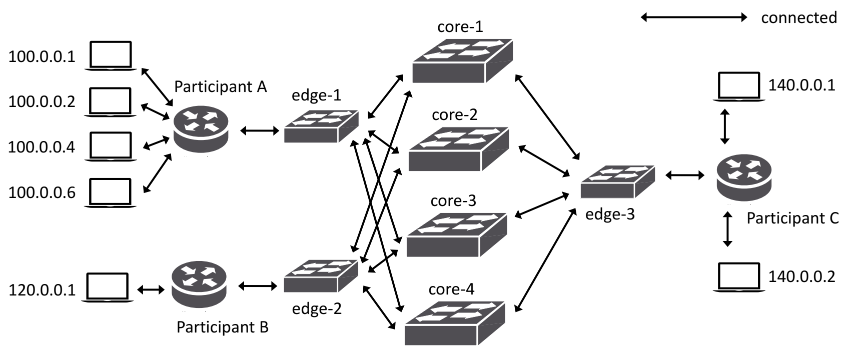


Figure 5: Topology used for the load balancing use case.

Table	**Priority**	**IP Src**	**IP Dst**	**Actions**	**Goto**
4	10	0.0.0.0/0.0.0.1	0.0.0.0/0.0.0.1	0x30	5
4	10	0.0.0.1/0.0.0.1	0.0.0.0/0.0.0.1	0x20	5
4	10	0.0.0.1/0.0.0.1	0.0.0.1/0.0.0.1	0x40	5
4	10	0.0.0.0/0.0.0.1	0.0.0.1/0.0.0.1	0x10	5

Table 1: Flow Rules example for switch edge-1

ing of packets on the edge switches as follows. On the edge switch packets can be forwarded onto different core switches. The decision for a particular core switch is done by using source and destination IP addresses of each packet as described in Table 1.

To demonstrate the effectiveness of the above Load Balancing simple function, five individual network flows are generated between several hosts of the members so as to simulate network traffic across the IXP infrastructure. We generate the following sequence of events within the simplified IXP topology and visualize the utilization of specific links at the IXP infrastructure in Figure 6, where the upper area of Figure 6 shows the network streams that are generated over the IXP topology while the lower area depicts the utilization of the four core switches, each in a different color, normalized as a percentage of the load on the IXPs core.

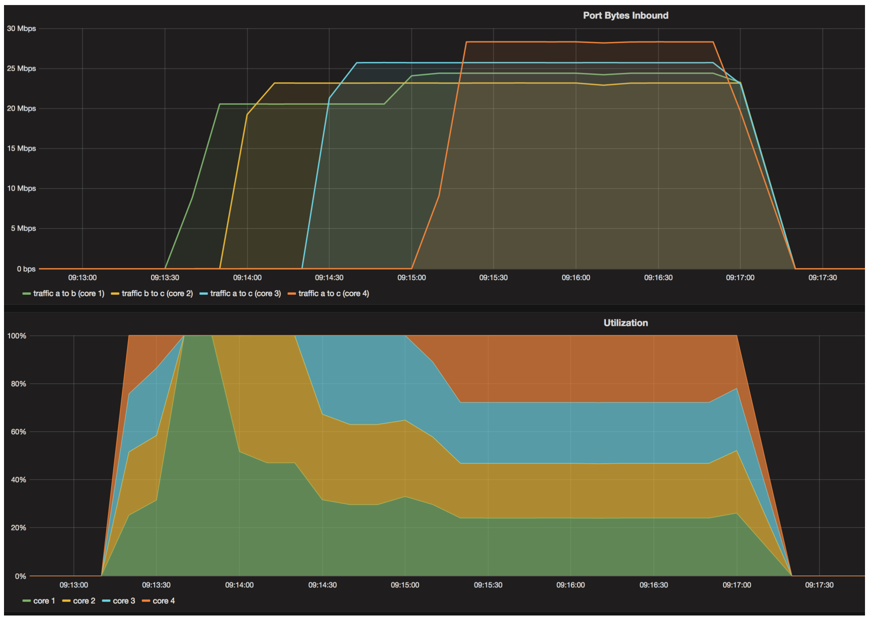


Figure 6: Traffic monitoring at core switchs.

1. Host `100.0.0.2` of member `A` starts network communication towards host `120.0.0.1` of member `B` at 09:13:30. Before that point in time only minor backscatter traffic was prevalent in the network, this is why the left-side of the bottom graph in Figure 6 shows a unclear pattern. With only one data stream, we can see that the traffic is solely transferred via `core-1`. This can be observed by the green area accounting for 100% of the traffic.

2. At 09:13:50 a second traffic flow is generated from host `120.0.0.1` of member `B` to host `140.0.0.2` of member `C`. This stream is forwarded through `core-2`.

3. One minute later, at 09:14:50, a third network communication is started from host `100.0.0.4` of member `A` to host `140.0.0.2` of member `C`. This traffic is processed by `core-3`.

4. Starting from 09.14.50, host `100.0.0.6` of member `A` adds a network communication towards host `120.0.0.1` of member `B`.

5. From 09:15:00 the fourth communication from host `100.0.0.1` of member `A` to host `140.0.0.1` of member `C` is initiated. This flow is for-

warded via `core-4`.

6. At 09:17:00 all network communications are stopped.

From the area graph of Figure 6 one can see the distribution of traffic over the IXPs core. As stated before, the mechanism exploits the pair of source and destination IP addresses as an identifier to forward the traffic to different cores. To be more specific, the least significant bit of the source and destination IP address is used as shown in Table 1. According to the combination of the least significant bits of the `IP Dst` and `IP Src` field, a different core is set as forwarding target, denoted by the `write meta data` in the `Actions` field. The `Goto` field points to the next table to be processed, which is then handled by the Umbrella component. Umbrella uses the layer 2 label switching mechanism to forward the packet to a specific core switch.

As stated before, the current load balancing mechanism can be extended in a flexible manner and additional algorithms can be developed and build on top of it.

A video that visualizes and demonstrates this use case can be found at:

- `https://www.youtube.com/watch?v=pChnqivN3A4`

Reproducing the use case demonstration. We leveraged *torch* for orchestrating and scheduling the network events needed to showcase the Load Balancing use case. The configuration file containing each network participant and each event is stored in `iSDX/test/specs/test3-mh-lb.spec`. We leveraged Grafana to visualize the flow of traffic through the IXP fabric. The web-based Grafana interface can be accessed via any browser installed on the Host machine by entering the following address `http://localhost:3000`. The visual dashboard for the Load Balancing use case is stored in the default Grafana setup.

5 Summary

In this report, we described the implemented ENDEAVOUR use cases for IXP operators. This document serves as an additional materials that accompany the videos of use cases' demonstrations. We outlined the setup of the development environment and we then presented the implemented use cases together with technical details. Finally, we discussed the workflow of each demonstrator.

6 Acronyms

SDN	Software Defined Networking
BGP	Border Gateway Protocol
ISP	Internet Service Provider
IXP	Internet eXchange Point
CDN	Content Delivery Network
IP	Internet Protocol
OSPF	Open Shortest Path First
HTTP	HyperText Transfer Protocol
VM	Virtual Machine
ARP	Address Resolution Protocol
ECMP	Equal-Cost Multi-Path Routing
MAC	Media Access Control
JSON	JavaScript Object Notation
SDX	Software Defined eXchange