SDEFIX - Identifying Elephant Flows in SDN-Based IXP Networks

Luis Augusto Dias Knob*, Rafael Pereira Esteves*†,
Lisandro Zambenedetti Granville*, Liane Margarida Rockenbach Tarouco*
*Institute of Informatics – Federal University of Rio Grande do Sul
Av. Bento Gonçalves, 9500 – Porto Alegre, Brazil
†Federal Institute of Rio Grande do Sul - Campus Restinga
Rua Alberto Hoffmann, 285 – Porto Alegre, Brazil
Email: {luis.knob, rpesteves, granville}@inf.ufrgs.br, liane@penta.ufrgs.br

***Abstract*—Internet Exchange Points (IXPs) are fundamental blocks of the Internet ecosystem by providing cost-effective connections among multiple autonomous systems (ASes). One of the main management challenges in IXP networks is the management of the so-called elephant flows. Elephant flows, characterized by high throughput and long duration, represent a small fraction of the total flows of an IXP network but have high impact on the overall traffic. A first step in the management of elephant flows is their identification, which becomes more important in SDN-based IXPs that require controllers to have a consistent view of the underlying network to allow fine-grained adjustment. In this paper, we propose, develop, and evaluate a monitoring system to identify elephant flows in an SDN-based IXP network. We demonstrate that our system can assist IXP operators to properly identify elephant flows in an efficient, scalable, and timely manner.**

***Index Terms*—Software Defined Networking, Internet Exchange Points, Network Management**

I. Introduction

Internet Exchange Points (IXPs) connect Internet autonomous systems (ASes) and enable service providers (SPs) to directly exchange traffic to better serve SPs' customers. IXPs are distributed worldwide and can serve dozen to hundreds of participants. According to a recent study [1], IXPs account for at least 20% of all traffic exchanged between ASes. Another important number shows that the daily aggregated traffic of a large IXP like DE-CIX can be compared with the traffic of a tier-1 ISP like AT&T [2]. One of the main benefits experienced by IXP participants is the low cost of deployment and maintenance [3]. As a consequence, companies such as Google [4], Facebook [5], and Netflix [6] seek to connect their infrastructures to major IXPs, thus reducing the costs of connecting customers to their services. Helping drive the Internet expansion for decades, IXPs are a critical building block of the global network.

IXPs enable the exchange of Internet traffic typically through an internal network of layer-2 switches [1]. The simplest IXP network is composed of a single switching device, while more complex IXPs can be formed by much larger networks with intricate topologies [2]. Not surprisingly, IXPs present management challenges that need to be tackled for the adequate IXP's operation [7]. That includes, for example, monitoring ARP traffic, establishing VPNs, fulfilling SLAs negotiated with participants, and dealing with critical flows and their associated network paths, which is central to this paper.

978-1-5090-0223-8/16/$31.00

Recently, with the development of Software-Defined Networking (SDN) technologies [8], a new set of opportunities started to be considered in the management of IXP networks. In that context, a typical SDN controller would be expected to manage the set of switches that form the IXP network. To take advantage of SDN, however, a complementary process that properly snapshots the IXP network status is required beforehand. With such snapshots, the IXP's SDN controller can tune the underlying network to meet its goals. In this paper, we are specially interested in identifying the so-called *elephant flows* traversing an IXP network. A flow is considered an elephant one if it has long duration and generates a significant amount of traffic [9]. Elephant flows also tend to be reduced in number but they account for most of the traffic in network infrastructures. In addition of identifying elephant flows in IXPs, we are also interested in discovering the paths that these flows use to traverse the IXP networks. Knowing these paths allows the IXP operator, for example, to identify links that form bottlenecks and, through the SDN controller, balance the network load by moving elephant flows from their original busier paths to more appropriate ones.

In our solution, we provide the IXP operator with a monitoring system that allows him/her to discover elephant flows and associated paths. That results in snapshots of the IXP network that can further feed, for example, an SDN controller that determines how flows are placed in the IXP network. Depending on the criteria used by the operator, he/she will be able in the future, by using an SDN controller, to decrease the blocking rate of new traffic, reduce IXP traversal delay, or improve energy savings. All these benefits, however, start with the adequate identification of elephants flows and associated paths, as addressed in this paper.

The remainder of this paper is organized as follows. In Section 2, we review related work on the management of

IXP networks. In Section 3, we present our proposed solution and report on the implementation of our prototype for the identification of elephant flows and associated paths in IXP networks. Our solution is then evaluated in Section 4, based on a network infrastructure inspired by AMS-IXP, which is an European IXP located in Amsterdam. We finally close this paper presenting concluding remarks and future work in Section 5.

II. Management of Internet Exchange Points

In this section, we review concepts that motivated our work. We first start by providing an introduction to IXPs. Then, we discuss management aspects of IXPs, focusing on the identification of elephant flows. Next, we review SDN in IXP management.

A. Internet Exchange Points

Internet eXchange Points (IXPs) are networking infrastructures that allow distinct Internet autonomous systems (ASes) to exchange traffic in a more efficient and cost-effective manner. By using IXPs, Internet Service Providers (ISPs) and Content Delivery Networks (CDNs) providers can exchange traffic without relying on third-parties, thus reducing the overall communication cost between ASes. IXPs evolved from the previous Network Access Points (NAPs) structures, which were deployed to orchestrate the transition from academic, government-sponsored networks (*e.g.*, NSFNET) to today's commercial Internet. Other similar initiatives included the Commercial Internet eXchange (CIX) and the Metropolitan Area Exchanges (MAEs). All these efforts resulted in the modern IXPs that are deployed worldwide [10] [11].

The commercial operation of IXPs can be divided into two separate groups: private and community. Private or "for-profit" IXPs are managed by a company that charges participants a monthly fee or based on the volume of traffic exchanged. On the other hand, community or "non-profit" IXPs are administered by the members themselves or by a non-profit organization. Typically, the model adopted by the IXPs across Europe, Africa, and Latin America is the community model, while in the USA and Canada there is a dominance of commercial IXPs. This happens both by commercial motivations (since its creation, the major American ASes have connections among one another) and legislative reasons, where some countries impose the traffic exchange between service providers operating in their territory [10].

Today, there are more than 300 active IXPs connecting thousands of ASes around the world [10]. IXPs are typically grouped according to their geographical location or the federation that they belong to. Examples of federations include Euro-IX (Europe) [12], LAC-IX (Latin America) [13], APIX (Asia-Pacific) [14], and Af-IX (Africa) [15]. The main goal behind federations is to promote and regulate the traffic exchange in these specific regions. All these federations are part of the global IX-F (Internet eXchange Federation). Reflecting their commercial nature, most IXPs from USA and Canada do not belong to any association. Recently, major Internet players such as Google, Netflix, and Akamai have started supporting the Open-IX Association [16]. The Open-IX Association defines a set of technical and operational standards for IXPs and emerged as a non-profit organization in North America that follows the model adopted in Europe.

B. Management of IXPs and Elephant Flows

Figure 1 depicts a infrastructure based on the AMS-IX [17], other IXPs tend to have similar structures. An IXP is typically composed of a number of layer-2 network switches or MPLS routers that allows ASes to exchange traffic. Each participating AS has to connect its access router to one of the switches/routers belonging to the IXP and establish a BGP session in order to peer with other ASes [11].

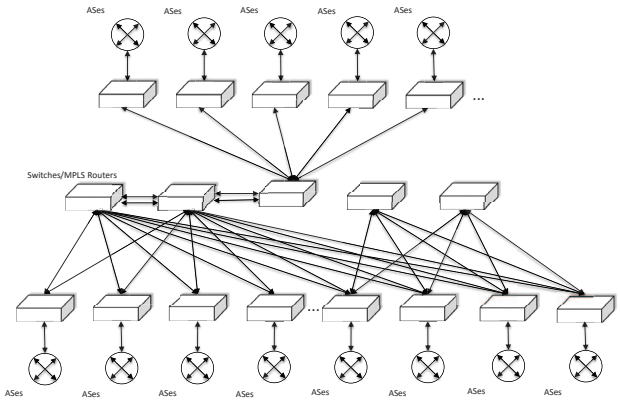

Fig. 1. An IXP infraestructure based on AMS-IX

The management of IXPs seeks the proper operation of the IXP infrastructure. For example, by using adequate monitoring tools, the IXP operator can have a better view of the IXP's network infrastructure to identify bottlenecks, handle failures, and make appropriate traffic engineering decisions to achieve better utilization of networking resources [18] [19]. Dealing with data confidentiality and promoting fair utilization of resources among ASes are other examples of IXP specific issues that need to be managed.

One IXP management aspect that has special importance is the management of *elephant flows*. In general, a flow is considered elephant if it has high throughput for a long period of time [20] [9] in comparison to other flows in the same network. Elephant flows account for a very small portion of the total flows of a network but can carry most of the traffic [9]. Managing elephant flows is important because of the potential impact of such flows in the IXP network. Elephant flows are long-lived flows that rapidly consume network buffers and impose undesirable queuing delays to short-lived ones, which are often referred to as mice flows. Therefore, mitigating the negative influence of elephant flows over mice flows and improving overall network utilization require appropriate management actions from the IXP operator.

The first step toward managing elephant flows in an IXP is to identify them. Once the IXP operator recognizes the elephant flows, he/she can take appropriate actions to optimize resource utilization and reduce the potential negative impact of elephant flows in the IXP network. For example, elephant flows can be isolated or migrated to other available network paths to make room for mice flows, use different routing strategies for elephant and mice flows, or use differentiated QoS schemes [9] [20].

C. SDN in IXPs

Software-Defined Networking (SDN) [21] [22] has been considered a viable alternative to tackle several issues in networking area. By decoupling the control plane from the forwarding plane, SDN empowers operators with full control of forwarding devices. Recently, the management of SDN and the use of SDN as a management tool have gained importance in the research community [8] [23].

In the context of IXPs, SDN can overcome the difficulties inherent to the Border Gateway Protocol (BGP), namely routing based on the destination prefix, limited application of policies (direct neighbors), and indirect path selection [24]. In this regard, SDN becomes a powerful tool to improve IXP management by delivering functionalities that were hard or even impossible before. These functionalities are realized by novel applications, including application-specific peering, inbound traffic engineering, wide-area server load balancing, and upstream blocking of DoS attacks [24] [25].

In this paper, we consider the situation of an SDN-operated IXP whose SDN logically centralized controller needs to be fed with information about the IXP's elephant flows. With such information, a set of SDN applications can optimize the IXP network according to some objectives, *e.g.*, reduce power consumption or decrease elephant flow traversal delay. Our goal is to snapshot the IXP traffic and, based on operators' requests, identify elephant flows and the path used by them to traverse the IXP network. We designed and implemented an IXP management system to achieve this goal, as presented in the next section.

D. Elephant Flows in SDN

There are several elephant flow identification strategies in SDN. A simple yet naive strategy is to define specific flow entries to capture elephant flows using an SDN standard protocol such as OpenFlow [26]. In this strategy, there are flow rules for identifying potential elephant flows in each switch of the network. These elephant flow rules are created using a flow abstraction (*e.g.*, OpenFlow). If a flow matches one or more elephant flow rules, then the flow is classified as an elephant flow. Other strategies were proposed in the context of SDN-based data center networks and include hierarchical statistics polling [27], host-based detection [28], and combination of sampling and triggering [29].

The problem with these strategies when applied to IXPs is threefold. First, they require collecting a large amount of traffic to allow the IXP operator to distinguish existing flows, which can slow down the identification process. Second, there should be as many flow entries as the number of potential elephant flows, which could exhaust the switch's flow table if the number of elephant flows to be identified is high. Finally, they can require modifications in the end-hosts or demand special hardware support that may not be possible in all IXPs.

In summary, current proposals for elephant flow identification are not well-suited for SDN-based IXPs. In the next section we introduce and detail our proposed monitoring system for SDN-based IXPs that overcomes the aforementioned limitations.

III. The SDN-based IXP Monitoring System

We developed a monitoring system named SDEFIX (Software-Defined Elephant Flow Identifier for Internet Exchange) that allows IXP operators to identify elephant flows and their associated network paths. We consider an environment where the IXP network is controlled by one logically centralized SDN controller, which runs applications that rule the underlying switches. Switches communicate with the controller through the OpenFlow protocol. Each OpenFlow switch maintains a flow table that is compared against incoming packets to determine the actions to be performed over those packets (*e.g.*, forward packet to a specific port) [26].

A. System Architecture

To overcome the scalability limitations of current elephant flow identification strategies in the context of IXPs, we designed a solution that uses sFlow [30] combined with OpenFlow. sFlow allows monitoring a large number of flows by employing flow *sampling*. Using sFlow, a sample packet is sent to a remote server, called *collector*, after a predefined number of packets is received by a network switch. In our solution, the collector stores samples in a NoSQL database (MongoDB). MongoDB provides a flexible structure to enable proper storage of packet samples. In MongoDB, the database schema may be dynamically modified according to the captured samples, which, in turn, allows storing information belonging to different network protocols. We use five MongoDB databases to store information regarding the identification rules, the collected flow samples, the flow tables of each switch, the network topology, and snapshots of identified flows, respectively.

Figure 2 depicts the architecture of our proposed system. It is composed of five main components and their respective databases: Elephant flow identifier, sFlow collector, Topology retriever, Flow Table retriever, and Administrative (Admin) interface. The Elephant flow identifier compares flow samples against identification rules and thresholds defined by the IXP operator through the Admin interface and returns the complete network path of each identified elephant flow. The sFlow collector collects flow samples in periodic intervals and sends the samples to the flow database. The Topology retriever and Flow Table retriever communicates with the SDN controller to gather the network topology and the flows tables of all

switches, and updates the topology and flow table databases, respectively. The Admin interface allows visualization of elephant flow paths and snapshots of previously identified flows, that may be used for historical record, or as a data source for IT infrastructure management. For each network switch, sFlow samples the traffic at specific polling intervals. We are employing the conventional interval of 10 seconds used in the sFlow's default configuration.

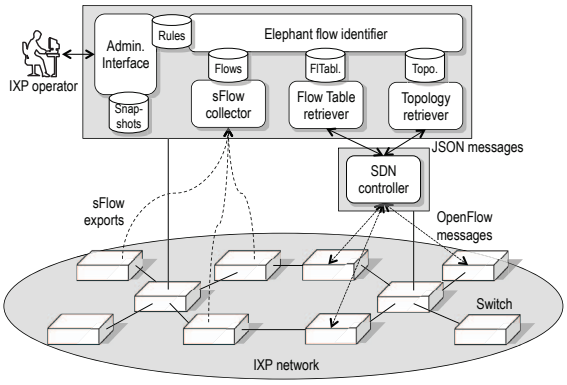

Fig. 2. SDN-based IXP Monitoring System

Although our solution is decoupled from any particular SDN controller, we are using the Ryu OpenFlow controller [31] in our implementation, since it was one of few controllers that supported latest OpenFlow versions and implements a well-defined REST API.

B. Elephant Flow Specification

In our solution, the IXP operator is responsible for defining what is an elephant flow. The IXP operator specifies identification rules to characterize elephant flows based on the following information: source/destination IP addresses, source/destination MAC addresses, transport protocol, and port numbers. The IXP operator can use multiple rules to identify elephant flows and only rules strictly written are recognized by the algorithm. Additionally, it is globally defined a bandwitdh threshold (in bytes/seconds) and a duration threshold (in seconds) for all flows. Figure 3 present a list of all input fields in the system, including the fields for new identification rules, the definition of the bandwitdh and duration thresholds and the sampling configuration with polling interval and sample rate.

Thresholds	Rules		
Bandwidth (in Bytes/Seconds)	Source MAC Address	Destination MAC Adress	
Duration (in Seconds)	Source IP Address	Destination IP Address	
Sampling Configuration	Transport Protocol (ICMP, UDP or TCP)		
Sample Rate (in Flows)	ICMP Type	Source UDP Port	Source TCP Port
Polling Interval (in Seconds)	ICMP Code	Dest. UDP Port	Dest. TCP Port

Fig. 3. SDN-based IXP Monitoring System - Input Fields

The identification rules are then used by the elephant flow identifier, which compares these rules with the active flows on the network, searching for potential elephant flow candidates. A detailed overview of this module is explained in the next subsection.

C. Elephant Flow Identification

Some works present solutions to identify elephant flows in SDN networks. The paper presented in [32] shows a solution for identifying elephant flows in composite structures with directing traditional and OpenFlow, checking via sampling packets above a dynamic threshold set by the average flow size on the network. After passing, the threshold is added to the controller rules to validate if that flow is actually an elephant flow. If confirmed, the flow then receives treatment in a separate controller. While at work [33], an identification solution is presented using sFlow, capturing network samples in order to classify flows with thresholds for UDP, TCP, and ICMP, using only the tool Inmon sFlow-RT [34]

Unlike other elephant flow identification approaches, our system allows the identification of the network paths associated with the elephant flows. Current solutions perform local, per switch flow identification, *i.e.*, the verification of the existence of elephant flows is performed at the switch scope, without knowledge of the other flows traversing the other switches of the infrastructure. The problem with that approach is that the IXP operator does not have a compiled list of devices through which an elephant flow passed along. Since in our system we keep track of the flow tables of all switches, we can match these flow tables with the identified elephant flows. If a switch has a flow entry corresponding to an elephant flow, we can include the switch in the path of the elephant flow. In this way, it is possible to derive the complete path for each identified elephant flow, helping the IXP operator to make appropriate management decisions with such flows.

Algorithm 1 illustrates the process of identifying elephant flows. The algorithm is triggered by the IXP operator when he/she wants to verify the elephant flows that are active in the IXP network at a given moment along with their associated network paths. In the elephant flow identification phase, the system compares the collected samples with the rules defined by the IXP operator (lines 9-11). If there is a match, the corresponding flow is added to the set of identified elephant flows (line 12). After that, the algorithm computes the network path associated with each identified elephant flow. This is performed by checking the flow table of each switch of the topology in order to find a flow entry that corresponds to the elephant flow (lines 17-21). If such an entry is found, the switch is included in the network path of the elephant flow (line 22).

As a result, the list of elephant flows matching the rules is then returned to the Admin Interface for a proper visualization by the IXP operator.

Algorithm 1 Elephant Flow Identification

R: set of rules defined by the IXP operator
S: set of collected flow samples
T: physical topology
$F(sw)$: flow table of the switch $sw \in T$
EF: set of identified elephant flows
$EFP(ef)$ network path of the elephant flow $ef \in EF$
Elephant Flow Identification:
$EF \leftarrow \varnothing$
for each $s \in S$ **do**
    for each $r \in R$ **do**
        if s matches r and s.bandwitdh $>$ *r.bandwitdh_threshold* and s.duration $>$ *r.duration_threshold* **then**
            add s to EF
        end if
    end for
end for
Elephant Flow Path Discovery:
for each $ef \in EF$ **do**
    $EFP(ef) \leftarrow \varnothing$
    for each $sw \in T$ **do**
        for each $f \in F(sw)$ **do**
            if $ef = f$ **then**
                add sw to $EFP(ef)$
            end if
        end for
    end for
end for

D. Displaying Elephant Flows Paths

Visualization and monitoring capabilities can assist IXP operators in network control and configuration tasks [35] [36]. In this perspective, SDNs require special attention, mainly because they present a high level of customization and programability. In this way, traditional visualization and monitoring solutions are not designed to cope with SDN requirements, mainly by the separation of the data and control plane and the necessity to manage both seamlessly. Therefore, an SDN-tailored monitoring/visualization solution can improve the understanding of the network behavior and simplify other management tasks [37].

After receiving the information from the Topology retriever and Elephant Flow Identifier modules, the Admin Interface returns to the user a visualization of the physical topology, as shown in Figure 4. The IXP operator can then visualize each identified elephant flow individually in the topology or can combine multiple elephant flows in a single view to highlight the most used network paths.

The operator can also generate a snapshot of the current network status and identified elephant flows for future analysis. Moreover, snapshots can be used to build a history of identified elephant flows and associated paths, and to assist the IXP operator in defining infrastructure update plans. When the system is initialized, the last saved snapshot is restored for the user.

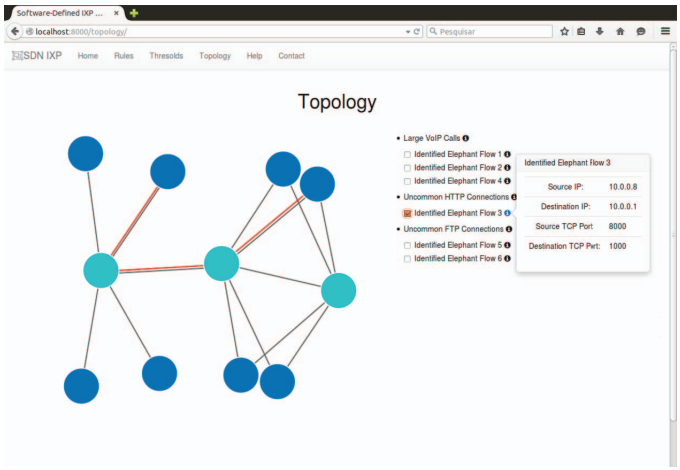

Fig. 4. Topology Visualization

In addition, the Admin interface was developed with the Django framework [38] that employs the MVT model (Model-View-Template) as a design pattern. Furthermore, the topology viewer was implemented using the D3js library [39], a powerful tool that uses JavaScript and SVG to create interactive visualizations.

IV. Evaluation

In this section, we evaluate the performance of SDEFIX in elephant flow identification. We begin by describing the experimental scenario and the methodology used in the tests. Following that, we present and discuss the main results achieved so far.

A. Scenario

The evaluation scenario is composed of three virtual machines (VMs) running inside a Core i7 4790 with 16 GB of RAM through VirtualBox. Each VM has 2GB and runs Debian 8 as operating system. The first VM runs the Mininet emulator [40] and contains the topology used in the tests. The topology, shown in Figure 5, is a simplified version the AMS-IX infrastructure [17] using SDN-enabled switches (AMS-IX uses MPLS routers). In our topology, we replace the AMS-IX core routers by 3 core switches and we simulate the connection of the IXP with 8 ASes using 8 edge switches instead of the PE routers used in AMS-IX. All switches are based on the OpenVSwitch [41] implementation. The second VM hosts the Ryu SDN controller that was chosen because it allows rapid development of network management applications. SDEFIX and its main components (Elephant flow identifier, sFlow collector, Flow table retriever, Topology retriever, Admin interface, and all the MongoDB databases) runs in the third virtual machine. For the sake of simplicity, we do not detail the internal topology of each AS. We consider that there are 32 hosts inside each AS generating traffic.

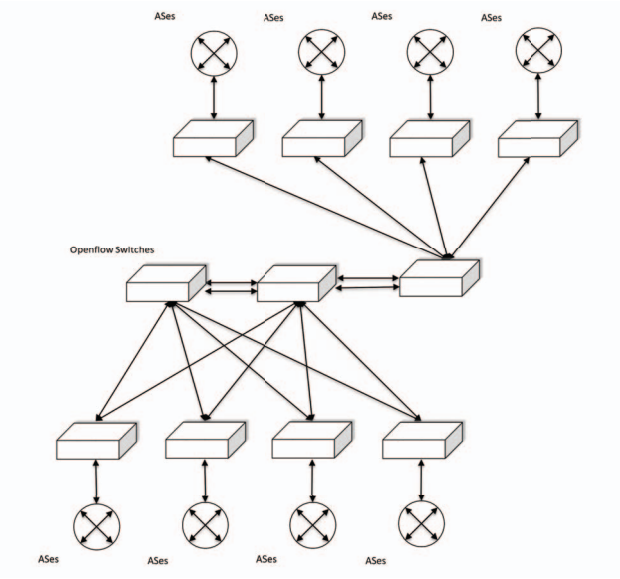

Fig. 5. Topology used in the experiments

In the experiments, the sFlow collector receives flow samples from the switches every 10 seconds, which is the conventional interval used in the sFlow's default configuration. The packet sampling rate varies from 1 in every 128 packets to 1 in every 4096 packets[1]. The size and duration thresholds used to characterize an elephant flow are 100Mbps and 10 seconds, respectively. Each host establishes 10 connections with other hosts in the network, 9 connections are used to characterize short (mice) flows with throughput between 4Mbps and 10Mbps, the remaining connection represents an elephant flow with throughput of 100Mbps. This means that there is a total of 128 elephant flows in the IXP network. The total number of flows (mice + elephant) is 1280.

We are interested in evaluating SDEFIX in terms of (1) *identification time*, (2) the *number of identified elephant flows*, (3) *resource usage*. The identification time is the time taken by the system to respond to a query executed by the IXP operator to identify the active elephant flows of the network at a given moment. The number of identified elephant flows is the number of elephant flows that were successfully identified by the system given a specific packet sampling rate. The sampling traffic is the amount of traffic resulting from the sampling process. Resource usage is defined in terms of CPU peak, memory usage, and database size. CPU peak and memory usage reflect the amount of resources consumed by SDEFIX in terms of CPU and memory, respectively. Database size is the sum of the size of all MongoDB databases used by SDEFIX. Each test was repeated 30 times with a confidence interval of 95%. In the next subsection, we present the main results of our evaluation.

[1]From this point, we use the notation $1/x$ to represent a packet sampling rate of 1 in every x packets.

B. Results

Figure 6 depicts the identification time when the number of rules used to identify elephant flows (identification rules) increases. In this case, the packet sampling rate is fixed to 1/128. Here, we compare two approaches for elephant flow identification. The first one, referred to as *sFlow identification*, is the case when the elephant flow and its corresponding path are identified when samples from that flow are collected from all switches of the path. That is, the path of an elephant flow can only be calculated if a sample of that flow is collected in all the switches of the path. In the second approach, called *2-step identification*, if samples of an elephant flow are collected from at least one switch, then the corresponding path can be calculated by matching the elephant flow with the entries in flow tables collected by the system using the strategy described in Algorithm 1. That is, if the elephant flow is found in a flow table of a switch, the switch is included in the path traversed by the elephant flow.

It is possible to observe from Figure 6 that 2-step identification performs in average 6% better than pure sFlow identification for a 1/128 sampling rate. Identification time remains stable around 0.6 seconds if the IXP operator queries 1 up to 32 elephant flows. For 64 and 128 identification rules, the identification time is 0.9 seconds and 1.8 seconds, respectively, which we believe is acceptable in realistic scenarios. Besides, the identification time increases in a slower rate compared to the number of identification rules. The identification time is three times higher (0.6 to 1.8) for a number of identification rules that is more than a hundred times higher (1 to 128), which represents a difference of two orders of magnitude.

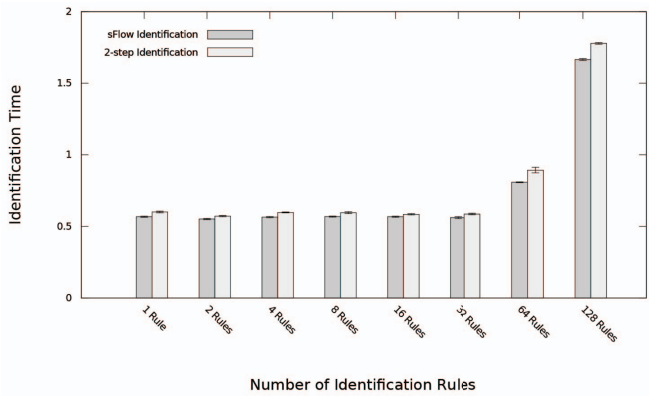

Fig. 6. Identification Time per Number of Identification Rules

We also measured the identification time for different packet sampling rates (1/4096, 1/2048, 1/512, 1/256, and 1/128 packets) when the number of identification rules is fixed in 128. We observed that the identification time does not vary significantly with the sampling rate, and the results are basically the same of Figure 6, which represents the worst case, *i.e.*, packet sampling rate of 1/128.

Figure 7 shows the average number of identified elephant flows when the packet sampling rate varies from 1/4096 to

1/128. It is possible to observe from Figure 7 that the number of identified flows increases linearly with the packet sampling rate for both identification approaches. This happens because the chances of collecting a flow sample that belongs to an elephant flow increase when the number of total flow samples is higher. The 2-step identification performs in average 20% better than pure sFlow identification. The difference between the two approaches can be explained by the fact that in sFlow identification there are some elephant flows that could not be identified in one or more switches of the IXP network, *i.e.*, there were no collected samples for those flows, and, as a consequence, the corresponding path could not be determined. On the other hand, in 2-step identification, the system needs that at least one switch sending a flow sample of an elephant flow to the sFlow collector. With a single flow sample, the system is able to determine precisely the corresponding path by comparing the flow sample with the flow entries of every switch of the IXP network.

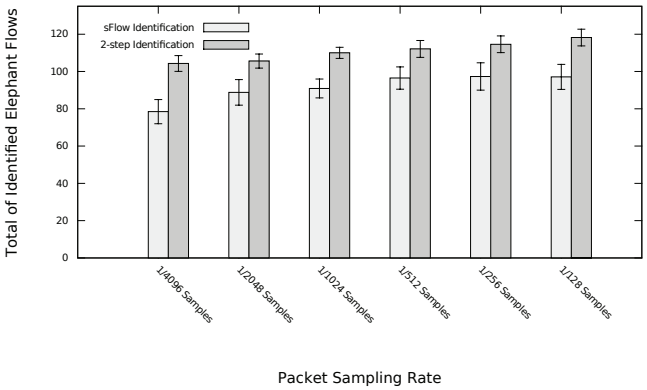

Fig. 7. Number of Identified Elephant Flows per Packet Sampling Rate

Table I shows the sampling traffic for different packet sampling rates. There is a significant difference (more than 20 times) in terms of generated traffic when a sampling rate of 1/128 (224 MB) is used, compared to a sampling rate of 1/4096, which generates only 11 MB of network traffic.

TABLE I
SAMPLING TRAFFIC (MBYTES)

Sampling rate	Traffic
1/128	224.4
1/256	101.2
1/512	61.6
1/1024	37.4
1/2048	19.1
1/4096	11

Table II summarizes the results regarding resource usage varying the number of identification rules (1 and 128) and the sampling rate (1/128 and 1/4096). CPU peak is mainly dependent on the number of identification rules and impacts the identification time as shown in Figure 6. The sampling rate has little influence in CPU usage. Memory usage remains stable around 155 MB. As expected, the sampling rate is directly related to the size of the database (*i.e.*, Flow Sample database) used by SDEFIX to store the collected samples.

TABLE II
RESOURCE USAGE

Metric	1/128 (128R)	1/128 (1R)	1/4096 (128R)	1/4096 (1R)
CPU peak	10.2%	4.3%	8.6%	2.8%
Memory usage (MB)	155	155	155	155
Database size (MB)	215.4	215.1	38.8	38.5

C. Summary

In summary, the identification time depends mainly on the number of identification rules defined by the IXP operator because, in the worst case, the system needs to compare flow samples with all the rules. Our 2-step elephant flow identification approach is capable of identifying in average 20% more elephant flows compared to the traditional pure sampling-based identification approach, which demonstrates the benefits of using an SDN-based solution for elephant flow identification.

As expected, there is a tradeoff between the precision of elephant flow identification and the amount of resources that are needed to allow such identification. The amount of generated traffic for a sampling rate of 1/128 is more than 20 times higher than the traffic generated by a sampling rate of 1/4096 packets. However, the precision gain with the 1/128 rate is only 13% better when compared to the 1/4096 rate (Figure 7). Resource usage in terms of CPU and memory indicates that SDEFIX is lightweight and can be deployed in commodity hardware. The main bottleneck is the database size, which is dependent on the sampling rate. Therefore, the IXP operator has to weigh both the accuracy of identification and the impact of the generated traffic and choose accordingly.

V. CONCLUSION

In this paper, we have proposed and presented SDEFIX, a monitoring system for SDN-based IXPs to allow identification of elephant flows and their associated paths. SDEFIX uses a sampling approach (sFlow) and leverages the flow table abstraction of the OpenFlow protocol to improve the precision of identification. Unlike most approaches for elephant flow identification based on sampling, SDEFIX does not require that every switch in the path traversed by an elephant flow sends a sample of that flow in order to calculate its path. Instead, in SDEFIX, only one sample is needed to allow the identification of an elephant flow and its associated path. We have evaluated SDEFIX in terms of identification time, number of identified flows, and sampling rate.

Evaluation results show that the identification time depends mainly on the number of identification rules defined by the IXP operator and is not significantly influenced by the packet

sampling rate. The packet sampling rate affects the accuracy of the identification, that is, the number of identified elephant flow increases with the sampling rate. However, improving the precision of elephant flow identification by using a higher sampling rate results in increased sampling traffic, which can affect the overall performance. Therefore, the IXP operator has to take both factors in consideration when choosing an adequate packet sampling rate. In resource constrained scenarios, it may be more important not to generate an excessive amount of sampling traffic, which can potentially affect other services. On the other hand, if precision of identification is crucial, an increase in sampling traffic can be tolerated.

Elephant flow identification is a first step towards the management of SDN-based IXPs. Such information can help IXP operators to define appropriate actions to handle such critical flows. As future work, we plan to carry additional experiments to evaluate SDEFIX considering different topologies and scenarios. We also intend to extend SDEFIX by taking the result of the identification as an input to adjust the network paths used by critical elephant flows.

REFERENCES

[1] J. C. C. Restrepo and R. Stanojevic, "IXP Traffic: A Macroscopic View," in *Latin American Networking Conference (LANC)*, Medellin, Colombia, 4–5 October 2012, pp. 1–8.

[2] B. Augustin, B. Krishnamurthy, and W. Willinger, "IXPs: Mapped?" in *ACM SIGCOMM Conference on Internet Measurement Conference (IMC)*, Chicago, USA, 4–6 November 2009, pp. 336–349.

[3] E. Gregori, A. Improta, L. Lenzini, and C. Orsini, "The Impact of IXPs on the AS-level Topology Structure of the Internet," *Computer Communications*, vol. 34, no. 1, pp. 68–82, 15 January 2011.

[4] Google. (2015) Peering Policy. https://peering.google.com/about/ peering_policy.html.

[5] Facebook. (2015) Peering Policy. https://www.facebook.com/peering/.

[6] Netflix. (2015) Peering Policy. https://www.netflix.com/openconnect/deliveryOptions.

[7] J. Matias, B. Tornero, A. Mendiola, E. Jacob, and N. Toledo, "Implementing Layer 2 Network Virtualization Using OpenFlow: Challenges and Solutions," in *European Workshop on Software Defined Networking (EWSDN)*, Darmstadt, Germany, 25–26 October 2012, pp. 30–35.

[8] J. Wickboldt, W. De Jesus, P. Isolani, C. Both, J. Rochol, and L. Granville, "Software-Defined Networking: Management Requirements and Challenges," *IEEE Communications Magazine*, vol. 53, no. 1, pp. 278–285, January 2015.

[9] L. Guo and I. Matta, "The War Between Mice and Elephants," in *International Conference on Network Protocols (ICNP)*, Riverside, USA, 11–14 November 2001, pp. 180–188.

[10] N. Chatzis, G. Smaragdakis, A. Feldmann, and W. Willinger, "There is More to IXPs Than Meets the Eye," *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 5, pp. 19–28, November 2013.

[11] B. Ager, N. Chatzis, A. Feldmann, N. Sarrar, S. Uhlig, and W. Willinger, "Anatomy of a Large European IXP," in *ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM)*, Helsinki, Finland, 13–17 August 2012, pp. 163–174.

[12] Euro-IX. (2015) European Internet Exchange Association. https://www.euro-ix.net.

[13] LAC-IX. (2015) Lac-ix (latin america & caribbean internet exchange association). http://www.lac-ix.org.

[14] APIX. (2015) Asia Pacific Internet Exchange Association. http://apix.asia.

[15] Af-IX. (2015) African Internet Exchange Association. http://www.af-ix.net.

[16] Open-IX. (2015) Open-IX Association. http://www.open-ix.org.

[17] AMS-IX. (2015) Amsterdam Internet Exchange Infrastructure. https://ams-ix.net/technical/ams-ix-infrastructure.

[18] Euro-IX. (2015) IXP Best Common Operational Practices. European Internet Exchange Association. https://www.euro-ix.net/documents/1391-euro-ix-ixp-bcops-221014-pdf.

[19] ISOC. (2009) Promoting the Use of Internet Exchange Points: A Guide to Policy, Management, and Technical Issues. Internet Society. http://www.internetsociety.org/promoting-use-internet-exchange-points-guide-policy-management-and-technical-issues.

[20] M. Casado and J. Pettit. (2013) Of Mice and Elephants. http://networkheresy.com/2013/11/01/of-mice-and-elephants/.

[21] W. Xia, Y. Wen, C. H. Foh, D. Niyato, and H. Xie, "A Survey on Software-Defined Networking," *IEEE Communications Surveys and Tutorials*, vol. 17, no. 1, pp. 27–51, Firstquarter 2015.

[22] ONF. (2015) Open Networking Foundation. https://www.opennetworking.org.

[23] R. P. Esteves, L. Z. Granville, and R. Boutaba, "On the Management of Virtual Networks," *IEEE Communications Magazine*, vol. 51, no. 7, pp. 2–10, July 2013.

[24] A. Gupta, M. Shahbaz, L. Vanbever, H. Kim, R. Clark, N. Feamster, J. Rexford, and S. Shenker, "SDX: A Software Defined Internet Exchange," *ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM)*, pp. 163–174, 17–22 August 2014.

[25] L. Vanbever, "Novel Applications for a SDN-enabled Internet Exchange Point," Berlin, Germany, SDN Research Group meeting, IETF 87, 29 July 2013.

[26] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: Enabling Innovation in Campus Networks," *SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, March 2008.

[27] C.-Y. Lin, C. Chen, J.-W. Chang, and Y. H. Chu, "Elephant Flow Detection in Datacenters using OpenFlow-based Hierarchical Statistics Pulling," in *Global Communications Conference (GLOBECOM), 2014 IEEE*, Dec 2014, pp. 2264–2269.

[28] A. Curtis, W. Kim, and P. Yalagandula, "Mahout: Low-overhead Datacenter Traffic Management using End-host-based Elephant Detection," in *Proceedings of IEEE INFOCOM 2011*, April 2011, pp. 1629–1637.

[29] A. R. Curtis, J. C. Mogul, J. Tourrilhes, P. Yalagandula, P. Sharma, and S. Banerjee, "DevoFlow: Scaling Flow Management for High-performance Networks," *SIGCOMM Comput. Commun. Rev.*, vol. 41, no. 4, pp. 254–265, Aug. 2011.

[30] sFlow. (2015) sFlow.org. http://www.sflow.org.

[31] Ryu. (2015) Ryu openflow controller. http://osrg.github.com/ryu.

[32] C. Bi, X. Luo, T. Ye, and Y. Jin, "On precision and scalability of elephant flow detection in data center with sdn," in *Globecom Workshops (GC Wkshps), 2013 IEEE*, Dec 2013, pp. 1227–1232.

[33] M. Afaq, S. U. Rehman, and W.-C. Song, "A framework for classification and visualization of elephant flows in sdn-based networks," *Procedia Computer Science*, vol. 65, pp. 672 – 681, 2015, international Conference on Communications, management, and Information technology (ICCMIT'2015). [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877050915028410

[34] InMon. (2015) sflow-rt. www.inmon.com/products/sFlow-RT.php.

[35] C. C. Machado, G. Tavares, L. Z. Granville, A. Schaeffer-Filho, and J. A. Wickboldt, "Towards sla policy refinement for qos management in software-defined networking," in *Advanced Information Networking and Applications (AINA), 2014 IEEE 28th International Conference on*, May 2014, pp. 397–404.

[36] V. Tavares, Guimaraes, G. L. dos Santos, G. da C. Rodrigues, L. Z. Granville, and L. R. Tarouco, "A collaborative solution for snmp traces visualization," in *Information Networking (ICOIN), 2014 International Conference on*, Feb 2014, pp. 458–463.

[37] P. H. Isolani, J. A. Wickboldt, C. Both, J. Rochol, and L. Z. Granville, "Interactive monitoring, visualization, and configuration of openflow-based sdn," in *Integrated Network Management (IM), 2015 IFIP/IEEE International Symposium on*, May 2015, pp. 207–215.

[38] D. Framework. (2015) Django: the Web framework for perfectionists with deadlines. https://www.djangoproject.com/.

[39] D3js. (2014) D3: Data Driven Documents. https://www.d3js.org/.

[40] Mininet. (2015) Mininet: An Instant Virtual Network on your Laptop (or other PC). http://mininet.org/.

[41] Openvswitch. (2015) Open vSwitch: Production Quality, Multilayer Open Virtual Switch. http://openvswitch.org/.