Software Defined Network Traffic Measurement: Current Trends and Challenges

Abdulsalam Yassine, Hesam Rahimi, Shervin Shirmohammadi
Distributed and Collaborative Virtual Environments Research Laboratory
University of Ottawa, Ottawa, Canada
{ayassine | hrahimi | shervin}@discover.uottawa.ca

1. INTRODUCTION

Next generation networks such as Software Defined Networks (SDN) must support the integration of new paradigms of service offerings such as virtualized cloud computing, big data applications, data centers services, and rich multimedia content. Operators of such networks are responsible to configure policies and employ traffic monitoring tools and measurement mechanisms to detect and react to a wide range of network events and applications. However, the huge scale and diversity of the generated traffic from these services make it difficult for network operators to measure, in short timescales, the status and dynamics of the network in effective and efficient ways [1]. In turn, this puts network operators in an unprecedented stress to satisfy users' expectation for delivering applications with guaranteed Quality of Service (QoS), because the fundamental prerequisite of offering guaranteed QoS is the presence of ubiquitous accurate traffic monitoring and measurement mechanisms. This is a very challenging prospect because many of today's applications (e.g. interactive media applications) require QoS to be maintained along the whole end-to-end network path, and therefore require capturing their momentous dependence on various factors of performance degradation. For example, to perform congestion detection for interactive real-time video conferencing applications [2], we need to accurately detect and measure bandwidth changes, resulting from cross traffic, in milliseconds to lessen quality degradation of the video. Furthermore, the ability to measure different types of network traffic at different time-scales is very critical for tasks such as traffic engineering and congestion detection to guarantee application performance. Today's network devices (e.g. switches, routers etc.) are inflexible to deal with different types of network traffic due to the underlying hardwired implementation of routing rules [3], and obstacles that SDNs promise to overcome. In this article, we take a look at traffic measurement methods is SDNs, cover their strengths and weaknesses, and point to open issues and remaining future challenges.

With the introduction of SDN [4] and OpenFlow [5] as novel approaches for better network management and virtualization, it is now easier to perform QoS measurement anywhere, anytime,

and anyplace through the use of self-directed, self-tuning mechanisms that continuously monitor and measure network performance and react swiftly to problems. The "globalization" of traffic measurement in SDN provides a seamless network management of complementary systems and complex applications consisting of context and QoS aware components. Using programmable interfaces to OpenFlow controllers, software-defined measurement solutions provide consistent traffic measurement of flow parameters, such as bandwidth, packet loss, and latency to support the diversity requirements of next generation network applications and services. The flexibility of software-defined measurement gives the network operator the capability to offer dynamic QoS that guarantees service quality between endpoints, regardless of what path that service is carried on. Furthermore, traffic measurement in SDN allows for an indirect, non-intrusive, and statistical way to infer several characteristics that in some cases cannot to be measured in traditional large networks. Before discussing these software-defined measurements, a brief overview of SDN and the OpenFlow protocol is needed, this is presented next.

2. OVERVIEW OF SOFTWARE DEFINED NETWORKS

The Open Networking Foundation (ONF), an organization dedicated to the promotion of SDN, defines SDN as the physical separation of the control plane from the forwarding plane (data plane) in traditional networks [4]. The data plane of the network typically consists of all the network elements (NEs) such as switches and the routers that allow packets to go from point A to point B. The control plane however, is the intelligence behind those NEs and decides how the packets should move from one NE to the other. Traditionally, when we look at any given network topology, there exists some very specialized hardware infrastructure that runs a dedicated (proprietary) operating system which has some specific applications or features. Users pay a lot of money for these closed boxes and they have to manage their networks by using those boxes. SDN is about breaking the above-mentioned closed environment by offering an open interface layer between the packet forwarding hardware and the network operating system that runs on the hardware, see Figure 1. By so doing, network management has been radically simplified. Decoupling the control plane from the forwarding plane enables the network control to become directly programmable and the underlying infrastructure to be abstracted for applications and network services [4]. In turn, this allows software developers to be oblivious of the underlying devices, and develop their networking logic the same way they do on any computing resources. Additionally, SDN architecture allows network administrators to dynamically adjust network-wide traffic flow to meet the changing needs of today's data centers, carrier environments and campuses.

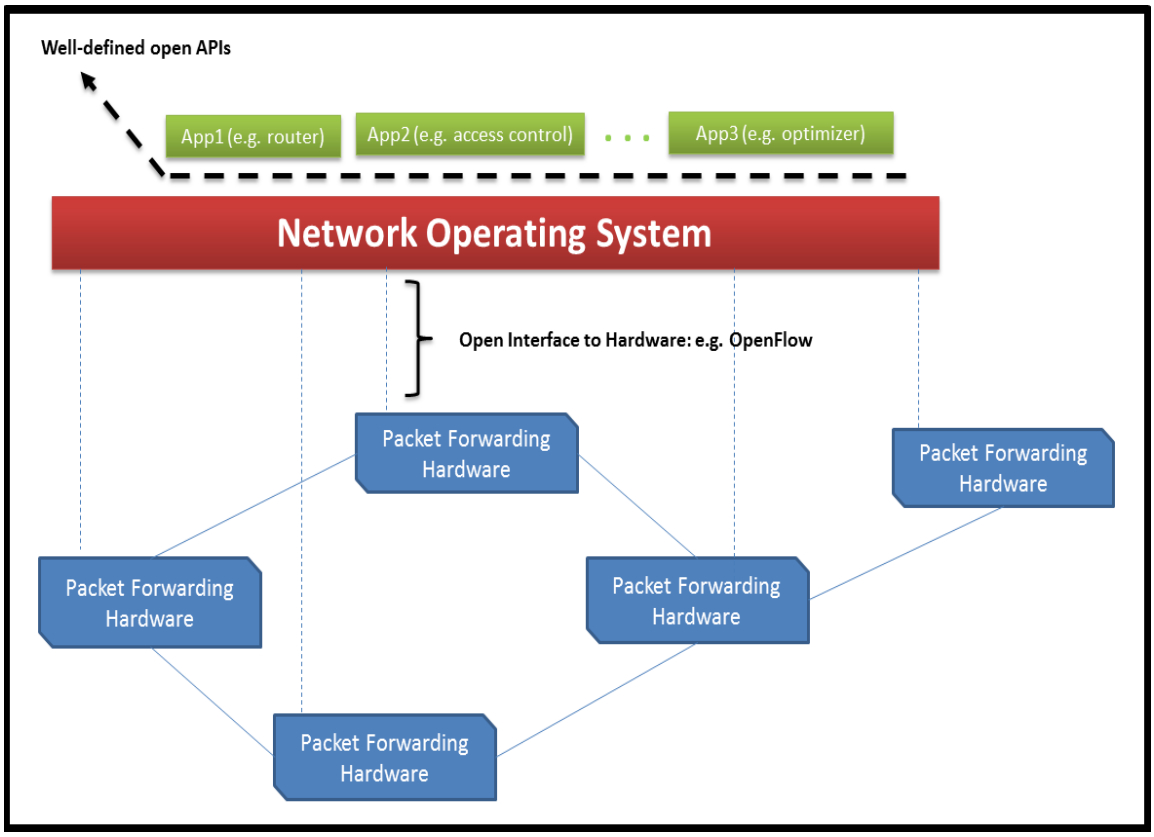

Figure 1: The concept of a software-defined network, adapted from [3]

The network intelligence in SDN is logically centralized in a piece of software called the "*controller*" or the "*network operating system*" as shown in Figure 1. This controller maintains a global view of the network and this view is accessible through some well-defined open APIs (typically called northbound APIs) to be used by different applications. These APIs do not depend on proprietary software or hardware, so network administrators can write those themselves and dynamically and automatically enforce some QoS policies to manage and control a large number of network devices and traffic paths. For example, the software developer can write APIs to interact with the network and acquire information from the switches about their status and program flow tables. In fact the spectrum of support that such APIs can provide is very wide and can be custom tailored to meet the business requirements of network operators.

One of the standardized protocols for the communication between the network operating system (the controller) and the packet forwarding hardware is the OpenFlow protocol [5]. Major vendors have already implemented this protocol in their commercially available OpenFlow-enabled network switches. An example of an OpenFlow-based switch is shown in Figure 2. This protocol defines messages such as packet-received, send-packet-out, modify-forwarding-table, and get-stats.

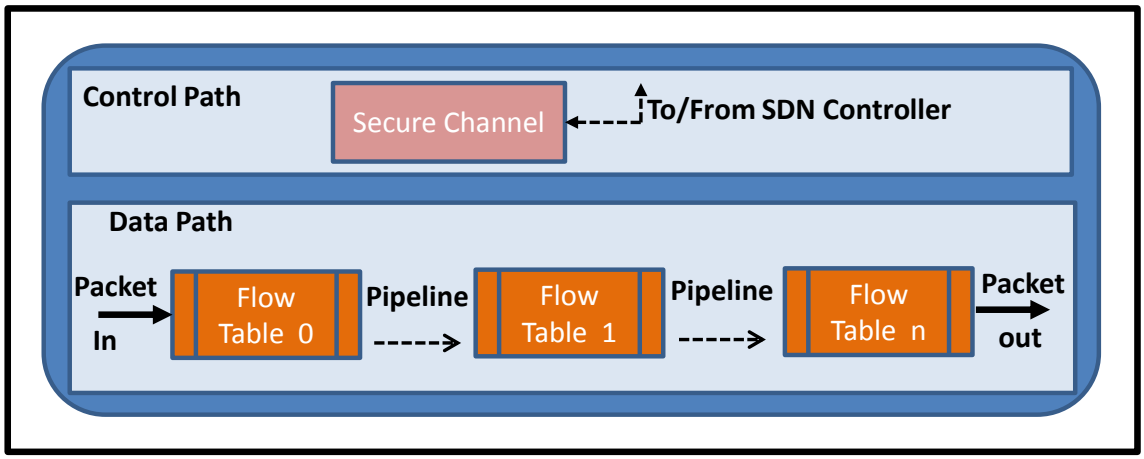

Figure 2: OpenFlow-based Switch, adapted from [5]

The data path of an OpenFlow switch presents a clean flow table abstraction; each row in this flow table consists of two parts: 1) match, and 2) action. A set of packet fields such as source or destination MAC addresses, Ethernet type, etc. will be used to match a specific flow, and an action (such as send-out-port, modify-field, or drop) will be set for each entry in this flow table. Once an OpenFlow switch receives a packet for which it does not have a matching entry (i.e. it has never seen that flow before), it sends this packet to the controller. It is now the controller's decision to handle this packet (to either drop, or forward to a specific port, and/or to modify the contents, etc.) and also to instruct the OpenFlow switches (by adding a flow entry) on what to do with similar packets coming in the future. OpenFlow is the only standardized SDN protocol that allows direct manipulation of the forwarding plane of network devices [4]. OpenFlow is currently being exploited and implemented by the research community and the industry in several applications related to network management, traffic measurement, network and data center virtualization and wireless applications [27]. For traffic analysis applications, OpenFlow allows for flexible automated fine-grained flow measurement, which makes it possible to develop innovative tools to improve traffic measurement capabilities of a switch using real time machine learning algorithms, databases, and any other software mechanism. These innovative software mechanisms will reduce operational cost, improve network stability, and support emerging IT services [4] .

3. CURRENT TRAFFIC MEASUREMENT APPROACHES IN SDN SYSTEMS

As discussed, next generation networks are characterized by their huge scale and diversity of the generated traffic. In such networks, it is not an easy task to predict the needed traffic measurement characteristics without enough measurement data about individual components in each part of the network. Traditionally, this is tackled by two types of measurement approaches, namely, active and passive methods [6]. In active measurement, network flows are continuously monitored for

performance, by sending special probe packets over the network paths. The aim of this process is, for example, to measure one-way delay, round-trip-time (RTT) or to adjust forwarding policies in response to load changes. Although active measurement can offer a different level of granularity of the end-to-end QoS measurement, it imposes significant measurement overhead that may disturb critical traffic flows. On the other hand, in passive measurement the network is tapped at predefined points where real traffic is captured and analyzed. Passive measurement does not cause any overhead in the network since it does not send any probe packets. It allows for processing of local traffic states as well as global behavior of traffic flows passing through specific network locations.

Active measurement in SDN requires careful planning to cope with the requirements of a centralized control architecture. The deployment of active measurement devices increases the data acquisition by multiple orders of magnitude, leading to SDN's centralized control mechanism to become quickly saturated [7]. Consequently, the streams of data acquired by the distributed devices in the network cannot provide the SDN controller with the necessary information in the timeframes necessary to minimize the impact of traffic disturbance [7]. Even in the presence of fast analytical models (e.g. machine learning techniques, Monte Carlo statistical methods, automation tools, etc.) aimed at converting the data into decisions, the controller faces communication bottlenecks, intractable control and optimization problems, increasing complexities, and vulnerability of the centralized infrastructures. Passive measurement methods, although non-intrusive and without generating additional traffic in the SDN, often depend on packet-sampling and employ statistical methods to infer the state of the traffic. The main issue with these techniques is that small flows may be missed or multiple monitoring nodes along an SDN flow path may sample exactly the same packet leading to measurement inaccuracies [8]. Furthermore, passive measurement requires sophisticated analytical mechanisms to process network traffic at high speed as in the case of today's datacenter networks.

Currently, several emerging approaches are trying to overcome the aforementioned challenges by efficiently utilizing the flexibility of SDN to offer programmable interfaces to attain fine-grained measurement of network flows. Existing studies, which are either proposing passive measurement methods or active measurement methods, can roughly be categorized into three main streams, see Figure 3: The first stream focuses on finding a balance between traffic measurement accuracy and overhead implications, by using techniques such as traffic sampling, aggregation, intelligent queries, etc. The second stream pays more attention to resources usage as a tradeoff with measurement accuracy. Finally, the third stream is mainly concerned with providing accurate traffic measurement in real time for reactive/proactive decision making.

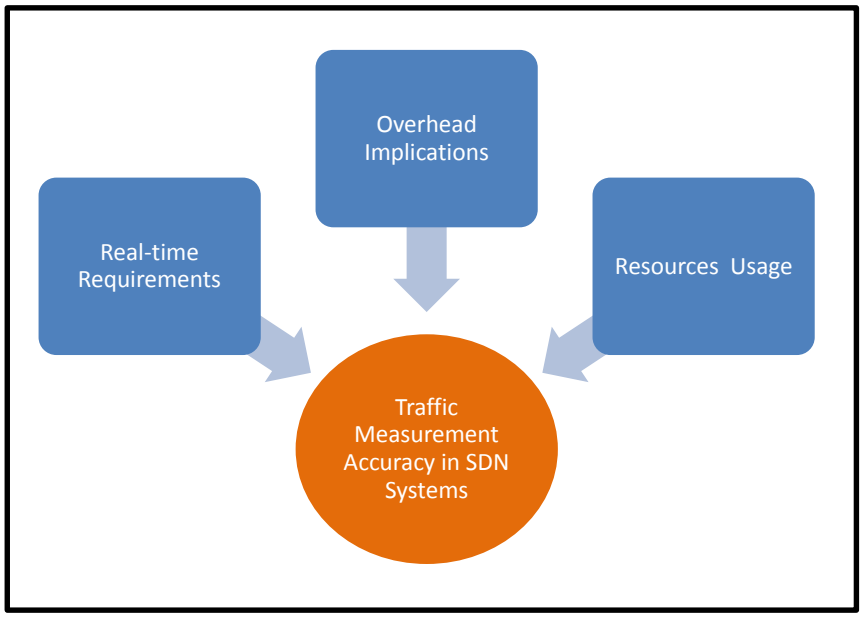

Figure 3: Traffic measurement in SDN

SDN Traffic Measurement Accuracy and overhead implications: Continuously monitoring the network often introduces overhead, which needs to be taken into consideration as a tradeoff with traffic measurement accuracy. In an effort to find a happy zone between accuracy and overhead implications, Jose et al. in [9] studied the prospect of measuring large scale traffic aggregation in commodity switches, by proposing a measurement framework where switches match packets against a small collection of wildcard rules available in Ternary Content Addressable Memory (TCAM). This approach reduces the overhead of the controller significantly, because the switch processing the packet identifies the matching rules locally and determines if it needs to drop the packet or forward it. The framework is evaluated using Hierarchical Heavy Hitter (HHH) program to understand the tradeoff between accuracy and overhead. However, updating matching rules is a major issue in this approach. Although the architecture proposes a separate controller to add new rules, lack of prior indication of HHH occurrence requires the installation of additional mechanisms to monitor and discover those HHHs. Similar to [9], OpenNetMon [10] and iSTAMP [11] leverage the means of OpenFlow to measure traffic parameters. OpenNetMon determines whether end-to-end QoS parameters are actually met for each flow. It is a pull-based active measurement approach where network flows are continuously monitored between predefined endpoints for throughput, packet loss and delay. OpenNetMon uses an adaptive fetching mechanism to pull data from switches where the rate of the queries increases when flow rates differ between samples and decreases when flows stabilize. iSTAMP uses (de)aggregation measurement mechanism, which dynamically partitions the TCAM entries to allow fine-grained or course measurement tasks of incoming flows. For example, one partition of TCAM is used for aggregated

flow measurement and another partition is for de-aggregation when direct per-flow measurement is required. Flows are "stamped" for direct measurement if they are deemed to be important. iSTAMP uses an intelligent Multi-Armed Bandit (MAB) based algorithm to process these two sets of measurements, which are then jointly processed to estimate the size of all network flows using different optimization techniques.

The tradeoff between measurement accuracy and overhead is also studied in [12] and [13]. The work in [12] proposes a framework that can instruct hash-based switches for collecting traffic information, along with the HHH algorithm for defining important traffic, to support different measurement tasks. However, monitoring rules need to be carefully delegated across the network. The work in [13] uses a prediction based algorithm for flow counting to detect anomalies because the granularity of measurement along both the spatial and the temporal dimensions changes dynamically. In this way, anomaly detectors can instruct the flow collection module to provide fine-grained measurement data in case of expecting an attack or it can collect coarser-grained flow data otherwise. The work in [14] proposes an OpenFlow-based approach, called OpenTM, for traffic matrix estimation using simple logic for querying flow table counters. The logic is based on keeping statistics for each active flow in the network. The information about active flows are pulled from the switches periodically and then compared for accuracy. Even though it follows the same concept as in FlowSense [16] to compute link utilization, OpenTM is an active network-wide measurement approach that at the end will introduce overhead in the process of periodically pulling statistical information from switches across the network. Furthermore, OpenTM uses a combination of selection methods to select switches for pulling information; this may lead to some measurement inaccuracy as investigated in [8]. In [15], an active monitoring framework for SDN, called Payless, is proposed. Payless focuses on the tradeoff between accuracy and network overhead. It provides a flexible RESTful API for flow statistics collection at different aggregation levels. The key advantage of Payless is that it uses an adaptive statistics collection algorithm to attain accurate information in real-time without incurring significant network overhead. It also uses Floodlight controller's API to implement the proposed mechanism. It has been shown through evaluation that Payless can indeed provide low overhead and can achieve higher accuracy of statistics collection than FlowSence. But, the accuracy seems to increase only at the expense of the overhead.

FlowSense in [16] uses passive measurement by which the network sends performance changes about the flows instead of querying the switches on demand. The approach utilizes the PacketIn and FlowRemoved messages, which are sent by switches to the controller when a new flow comes in or upon the expiration of a flow entry. These capabilities are provided by OpenFlow to query

switches for the number of packets or bytes in flows matching against a specific rule or traversing a network link. Evaluation results show that FlowSense has a promising performance compared to other approaches and can accomplish 90% of link utilization under 3 seconds.

SDN Traffic Measurement Accuracy and resources usage: The above measurement techniques were not concerned by resource usage and its effect on measurement accuracy. DREAM [17] is a dynamic resource allocation software-defined measurement framework that balances between user-specified level of accuracy and resource usage for measurement activities. In DREAM, resources are not allocated before the execution of the measurement task, but are dynamically deployed to achieve the desired level of accuracy based on traffic characteristics. DREAM framework is tested using HHHs programs show that DREAM can support more concurrent tasks with higher accuracy than several other alternatives. Aligning measurement tasks between the host and the network is also a major activity that in the end may reduce the overhead of an active measurement approach. The work in [18] argues that current controller applications in SDN systems are designed to be proactive, which may require the switches to accommodate a number of flow table entries that exceed the capabilities of their TCAMs. While equipping SDN switches with more powerful TCAMs is a feasible option, this may come at the expense of increasing operation and power consumption cost. The study proposes that controllers should consume resources efficiently using a reactive logic control approach. As in DREAM, the study suggests that resources must be allocated and freed depending on the network load, the effective behavior of the flows, their granularity and their inter-packet arrival time. The evaluation of the system shows that such approach is promising to enhance the traffic measurement flexibilities without extending the flow tables. Another algorithm called Baatdaat is studied in [19] and uses OpenFlow running on NetFPGA programmable switches, which permits real-time dynamic flow scheduling in datacenters. The proposed algorithm can adapt to instantaneous traffic bursts as well as to average link load by using spare DC network capacity to mitigate the performance degradation of heavily utilized links. Experiments show that Baatdaat can reduce network-wide maximum link utilization by up to 18% equal cost multipath (ECMP).

Finally, the proposed HONE platform in [20] presents a uniform stack for a diverse collection of measurements in SDN-based systems. HONE uses software agents residing on hosts, and a module interacting with network devices. Since continuously collecting statistical data about network flows is expensive, HONE offers two techniques to process flow statistics: The first technique, known as lazy materialization of the measurement data, uses database-like tables for uniform abstract representation of statistical data collected from hosts and network devices. The aim of this technique is to minimize measurement overhead by allowing the controller and the host agents to

analyze queries of necessary statistics for multiple management tasks at appropriate frequencies. The second technique offers data parallel streaming operators for programming the data-analysis logic. The operators can also be used in a hierarchically fashion for aggregate analysis among multiple hosts. Scalability is a main problem in deploying HONE as software agents need to be installed in every host and then synchronized to populate the statistical tables in a timely fashion to process meaningful queries.

SDN Traffic Measurement Accuracy in real time: Traffic measurement in SDN extensively relies on collecting statistical data about network flows in real time. The large amount of detailed statistics provided by the hosts may raise a scalability issue for real-time analysis, specifically when the measured data is for time sensitive applications. The study in [21] proposes PLANCK, a software-defined measurement architecture, which utilizes the capability of port mirroring that exists in most commodity switches. The extracted measurement statistics of the network flow is achieved in 280 microseconds to 7 milliseconds on a 1 Gbps commodity switch and 275 microseconds to 4 milliseconds on a 10 Gbps commodity switch. This is faster than in traditional networks by orders of magnitude. Port mirroring is a common approach to monitor traffic passing through the mirror using a variety of network analyzers and security applications. But while it is a useful technique, traffic volume may exceed the capacity of the ports, causing the switch to start dropping packets. To solve this issue the authors suggest buffering the traffic statistics for further analysis. However, the process will only be useful to study disturbing events after the fact and not to take actions to remedy situations in real time. Using sampling-based SDN measurement methods, IBM research lab proposes OpenSample [22], which leverages sFlow [23] packets to provide near–real-time measurements of both network load and individual flows by capturing packets from the network. OpenSample is a low-latency platform that uses TCP sequence numbers from the captured headers to measure accurate flow statistics. Using sampling for network monitoring allows OpenSample to have a 100 millisecond control loop rather than the 1–5 second control loop of traditional polling-based approaches. It is implemented with Floodlight OpenFlow controller and evaluated on a testbed comprised of commodity switches. One of the main advantages of OpenSample is that it considers any TCP flow , hence it can detect elephant flows (very large and continuous flows), and requires no modification to switches, making it highly marketable.

As an alternative to OpenFlow, OpenSketch [24] is proposed as a software-defined measurement architecture. OpenSketch uses a measurement library in the control plane to automatically configure and manage resources for measurement activities. The library makes it easier to customize and apply theoretical algorithms to measure flows in commodity switch

components. OpenSketch can be used for several measurement activities including HHH measurement, traffic distribution and link utilization. The main barrier for OpenSketch as a marketable SDN traffic measurement solution is the need for upgrading network nodes, which is a very expensive undertaking. Furthermore, it is very rigorous and time consuming to standardize a new protocol. OpenFlow has already taken off and is widely accepted as an industry standard in datacenter environments and is increasingly being implemented in commodity switches. With OpenFlow gaining momentum, it will be faster adopted by ISPs and research communities.

Table 1, summarizes the discussed approaches of traffic measurement in SDN. It must be mentioned that several works such as network monitoring, flow management, fault tolerance, and topology update, although beyond the scope of this article because they are not specifically related to traffic measurement, are equally important for the wider sense of traffic engineering in SDNs, and can be studied in [25] and [26].

Table 1: Comparison of current traffic measurement methods in SDNs

Method	**Measurement Type**	**Mechanism**	**Analysis Summary**
Jose et al. [9]	Active	Measure large scale traffic aggregation using switch matching rules	Reduces the overhead, but requires continuous update to the matching rules
OpenNetMon [10]	Active	Adaptive fetching of data from switches.	Accuracy increases at the expense of overhead
iSTAMP [11]	Active	TCAM partitioning for aggregate and de-aggregate traffic.	Accuracy increases, however it needs additional mechanism to "stamp" important flows.
Moshref et al. [12]	Active	Instructs hash-based switches for collecting traffic information.	Increases accuracy, but monitoring rules need to be carefully delegated across the network.
Zhang [13]	Active	Prediction based algorithm for flow counting to detect anomalies.	Accurate for identified traffic only.
OpenTM [14]	Active	Constantly polling the switch for collecting flow statistics.	High accuracy and high overhead.
Payless [15]	Active	Adaptive algorithm for polling flow statistics with low and high interval frequency.	Varying accuracy and overhead based on the length of the polling interval.
FlowSense [16]	Passive	Utilizes PacketIn and FlowRemoved.	High accuracy and low overhead.
DREAM [17]	Active	Resources are dynamically deployed to achieve the desired level of accuracy.	Higher accuracy for concurrent tasks compared with other alternatives.
Dusi et al. [18]	Active	Argues for equipping SDN switches with more powerful TCAMs.	Increases accuracy at the expense of adding resources.
Baadaat [19]	Active	Scheduling algorithm	Adapts to instantaneous

		for real-time dynamic flow scheduling in datacenters.	traffic bursts as well as to average link load.
HONE [20]	Active	Uses software agents residing on hosts, and a module interacting with network devices for periodically collecting statistical data about network flows.	Scalability is a main problem in deploying HONE as software agents need to be installed in every host and then synchronized to populate the statistical tables in a timely fashion.
PLANCK [21]	Active	Uses port mirroring that exists in most commodity switches.	Very fast statistical results, but traffic could exceed the capacity of the ports.
OpenSample [22]	Passive	Using sFlow and TCP sequence numbers for achieving low latency.	High accuracy with low latency.
OpenSketch [2 4]	Active	Uses a measurement library in the control plane to automatically configure and manage resources for measurement activities.	High accuracy with low latency.

4. CHALLENGES AND OPEN RESEARCH ISSUES

There is no doubt that SDN is a breakthrough in the networking industry as we know it today. Traffic measurement is a key enabler to achieve the potential benefits of the SDN paradigm; however, there are still challenges and several critical research issues. In Figure 4, we provide an overview of four major challenges of SDN traffic measurement and the road map for further research.

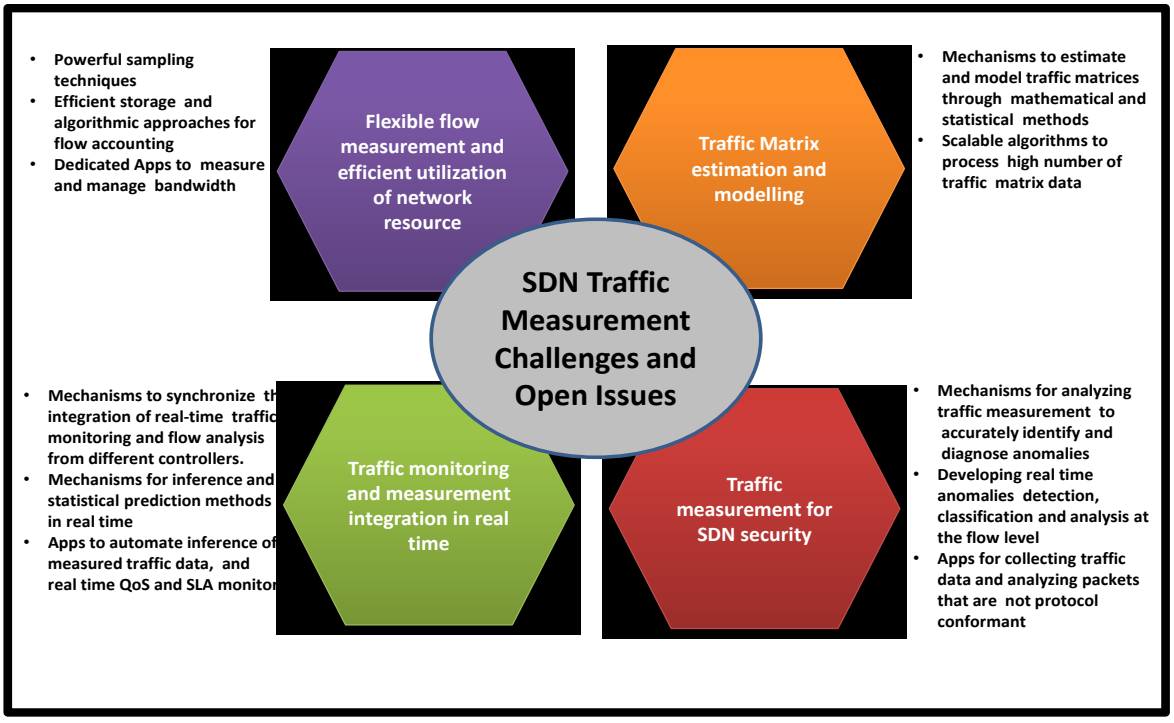

Figure 4: Challenges and open issues of traffic measurement in SDN

Flexible flow measurement and efficient utilization of network resources: The main goal of traffic measurement in SDN is to provide a flexible flow measurement with different granularities to satisfy a variety of applications. This task; however, is not trivial because it requires estimation of fine-grained volume of network flows with flexible settings in interconnected heterogeneous large scale networks. In addition, it requires traffic prediction, forecasting, and estimation of network resources. There are some scenarios where over time, network resources are not used properly i.e. link utilization/bandwidth utilization becomes non-optimal over time in the network due to scattered network services. For example, when new services with small bandwidth are being provisioned without consideration of future services that might possibly require more bandwidth, then the most optimum routes will get assigned to services with small bandwidth (first-come first-serve). Later, when we want to provision high-bandwidth services, inevitably less optimum paths will be used for them. In such scenarios, mechanisms are needed to provide flexible and dynamic measurement schemes that would inform network operators and help them maximize their resource utilization, rearranging the network by allocating small services to another path dynamically, and better utilize link capacities for the new high-bandwidth services. To do such rearrangement with minimum impact/overhead, it requires research in powerful sampling techniques to learn about flows and estimating their size, efficient storage, algorithmic approaches for flow accounting, and dedicated APIs that run on controllers to measure and manage bandwidth.

Traffic matrix estimation and modelling: Traffic matrices reflect the amount of traffic that flows between pairs of sources and destination in a network. The OpenFlow protocol in SDN provides flow table counters that allow other mechanisms to estimate traffic matrices, by querying the flow tables as shown in [13]. The information provided by traffic matrices (e.g. size of flows) are of special importance for diagnosing traffic related problems, discovering traffic anomalies, routing, and load balancing. Next generation networks are complex, heterogeneous, and with large-scale interconnected domains. Therefore, it is very challenging to accurately determine the traffic matrix when services are provisioning dynamically in the network. The logic for forwarding the packets is determined by the SDN controller and is implemented in the flow table at the SDN forwarding elements. Providing traffic matrices in a ubiquitous fashion to be used for optimal traffic engineering in SDN is still in its infancy stage. There is a pressing need to find mechanisms to estimate and model traffic matrices through mathematical and statistical methods as well as scalable algorithms to process high number of traffic matrix data.

Traffic monitoring and measurement integration in real time: Real time applications are delay sensitive and demand stringent QoS requirements and provisioning to adapt to network changes and dynamic resource allocation. For example, wireless mobile applications require real time traffic

monitoring and measurement to adapt to network channel changes due to users' mobility. Other examples such as online multi-player games, interactive online learning applications etc. connect very large number of users who interact with the application and response to each other in real time. These applications fundamentally require QoS to be maintained along the whole end-to-end network path based on a global network-wide policy. Although SDN allows the real-time centralized control of a network and user defined policies, the planner must deal with scalability issues to integrate significant number of fine-grained measurement statistics to a centralized controller for quick decision to adapt to QoS policies. Decision making at the logically centralized controller needs mechanisms to synchronize the integration of real-time traffic monitoring and flow analysis from different controllers. They also need mechanisms for inference and statistical prediction methods in real time to compensate for size limitation of the controller back-end database. Special focus would be on constructing APIs that run on the controller to do automated inference of measured traffic data, context dependent traffic analysis, real time QoS and Service Level Agreement monitoring (e.g. end-to-end delay) for delay-sensitive applications.

Traffic measurement for SDN security: As SDNs become widely deployed in cloud computing infrastructure, data centers, carrier networks and other highly sensitive computing paradigm, potential security vulnerabilities in the form of external and internal attacks are expected to rise. As such, integrated traffic measurement and application monitoring architectures for network security are most needed to detect and combat attacks on valuable assets. Currently, there has been limited traffic measurement approaches for SDNs considering security issues. As a matter of fact, SDNs are more vulnerable to attacks than traditional networks. This is mainly due to the centralized architecture of SDN, which makes the controller an appealing target for attacks to control the operation and carry out malicious activities inside the entire SDN. Other forms of attacks such as Denial of Service allows the adversary to reach individual network nodes, hosts, or users, and undermine the desired network performance. Traffic measurement mechanisms are therefore required to diagnose specific sources of events, security violations and attacks in the SDN. Traffic measurement data can be harvested from nodes and then analyzed and matched against security policies to reduce the possibilities of misconfiguration that opens doors for attackers. Mechanisms for analyzing traffic measurement data are also needed to accurately identify and diagnose anomalies as well as isolate and trace back anomalous signals within that data. However, when the controller itself is under attack (or the source of the attack), then the open question becomes: where to correlate detected events of an attack and how to analyze and coordinate a fight back? The answer to this question may be in developing real time anomalies detection, classification and analysis at the flow level in SDN switches. Also, this may require developing applications for

collecting traffic data and analyzing packets that are not protocol conformant for quickly identifying unusual behaviour.

5. CONCLUSION

In this article, we presented the current trends in traffic measurement in SDNs. Specifically, we covered important proposed mechanisms and highlighted their strengths and weaknesses. In general, traffic measurement in SDN is still in its infancy stage and several challenges require further research. We have identified those challenges and pointed out some important unsolved research issues, which require further in-depth investigations.

REFERENCES

[1] F.P. Tso, D.P. Pezaros . "Improving Data Centre Network Utilisation Using Near-Optimal Traffic Engineering" IEEE Transactions on Parallel and Distributed Systems (IEEE TPDS), Vol. 24(6), pp. 1139-1148, June 2013

[2] A. Javadtalab M., Semsarzadeh, A. Khanchi, S. Shirmohammadi, A. Yassine, "Continuous One-Way Detection of Available Bandwidth Changes for Video Streaming Over Best-Effort Networks," IEEE Transaction on Instrumentation and Measurement, 2014 doi: 10.1109/TIM.2014.2331423

[3] H. Kim and N. Feamster, "Improving network management with software defined networking," IEEE Communications Mag., Vol. 51, No. 2, pp. 114-119, 2013

[4] Open Networking Foundation "Software Defined Networks: The new Norm of Networks" White paper 2012 Available at: https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf

[5] OpenFlow, http://www.openflow.org/

[6] V. Mohan, Y.R. J. Reddy, K. Kaplan " Active and Passive Network Measurements: A Survey" International Journal of Computer Science and Information Technologies, Vol. 2 (4), pp 1371-1385, 20111

[7] S. Sezer et al., "Are We Ready for SDN? Implementation Challenges for Software-Defined Networks," IEEE Commun. Mag., vol. 51, no. 7, July 2013, pp. 36–43

[8] M. Jarchel, T. Zinner, T. Hohn, PTran-Gia " On the Accuracy of Leveraging SDN for Passive Measurements" Australian Telecommunication Networks and Applications Conference ATNAC, 2013

[9] L. Jose, M. Yu, and J. Rexford, "Online measurement of large traffic aggregates on commodity switches," in Proc. of the USENIX HotICE workshop, 2011

[10] N. L. M. van Adrichem, C. Doerr, and F. A. Kuipers, "OpenNetMon: Network monitoring in Oenflow software-defined networks," in Network Operations and Management Symposium (NOMS), 2014 IEEE

[11] M. Malboubi, L. Wang, C.N. Chuah, P. Sharma " Intelligent SDN based Traffic (de)Aggregation and Measurement Paradigm (iSTAMP)" IEEE INFOCOM 2014

[12] M. Moshref, M. Yu, R. Govindan, Resource/accuracy tradeoffs in software-defined measurement, in: Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking, HotSDN '13, August 2013, pp. 73–78

[13] Y. Zhang "An adaptive flow counting method for anomaly detection in SDN" In Proc. of ACM CoNEXT, 2013

[14] A. Tootoonchian, M. Ghobadi, and Y. Ganjali, "OpenTM: traffic matrix estimator for openflow networks," in Passive and Active Measurement.Springer, 2010, pp. 201–210

[15] S.R. Chowdhury, M.F. Bari, R. Ahmed, R. Boutaba, "Payless: a low cost network monitoring framework for software defined networks" Proceedings of the 14th IEEE/IFIP Network Operations and Management Symposium, NOMS'14, May 2014

[16] C. Yu, C. Lumezanu, Y. Zhang, V. Singh, G. Jiang, H.V. Madhyastha, "Flowsense: monitoring network utilization with zero measurement cost" Proceedings of the 14th International Conference on Passive and Active Measurement, PAM'13, March 2013, pp. 31–41

[17] M. Moshref, M. Yu, r. Govindan, A. Vahdat " DREAM: Dynamic Resource Allocation for Software-defined Measurement" ACM SIGCOMM, Chicago, Illinois, August 2014

[18] M. Dusi, R. Bifulco, F. Gringoli, F. Schneider, "Reactive logic in software-defined networking: Measuring flow-table requirements," Wireless Communications and Mobile Computing Conference (IWCMC), 2014 International , vol., no., pp.340,345, 4-8 Aug. 2014

[19] F. P. Tso, P. D. Pezaros, "Baatdaat: Measurement-based flow scheduling for cloud data centers," Computers and Communications (ISCC), IEEE Symposium on , pp.765,770, 7-10 July 2013

[20] P. Sun, M. Yu, M.J. Freedman, J.Rexford, D. Walker "HONE: Join Host-Network Traffic Management in Software-Defined Networks" Journal of Networks and Systems Management 10.1007/s10922-014-9321-9, 2014

[21] J. Rasley, B. Stephens, C. Dixon, E. Rozner, W. Felter, K. Agarwal, J. Carter, R. Fonseca " Planck: millisecond-scale monitoring and control for commodity networks" SIGCOMM'14, August 17–22, 2014, Chicago, IL, USA

[22] J. Suh, T. Kwon, C. Dixon, W. Felter, and J. Carter, ''OpenSample: A low-latency, sampling-based measurement platform for sdn,'' IBM Research Report, January 2014

[23] sFlow www.sflow.org

[24] M. Yu, L. Jose, R. Miao "Software defined traffic measurement with Opensketch" Proceedings of the 10th USENIX Symposium on Networked Systems Design and Implementation, NSDI'13, vol. 13, April 2013, pp. 29–42

[25] F. Hu, Q. Hao, and K. Bao, "A survey on software defined networking (SDN) and openflow: From concept to Implementation," IEEE Communication, Surveys Tutorials., no. 99, pp. 1–1, 2014

[26] I.F. Akylidiz, A. Lee, P. Wang, M. Luo, W. Chou " A Roadmap for Traffic Engineering in SDN-OpenFlow Networks" Elsevier Computer Networks, Volume 71,4 pp 1-30, 2014

[27] A. Lara, A. Kolasani, B. Ramamurthy "Network Innovation using OpenFlow: A Survey" IEEE Communication, Surveys Tutorials, Vol. 16, No. 1, pp 493-512, 2014