Towards a Shared Evaluation Environment for Software-defined Networking

Addis Dittebrandt, Michael König and Felix Neumeister [1]

Abstract: The evaluation of Software-defined Networking control plane applications is a complicated task with many pitfalls. This paper introduces the vision of a shared evaluation environment for Software-defined Networking (called SEED), setting out to ease and speedup the process of designing, setting up and replicating simulative evaluations for both the researcher and reviewer. SEED introduces an additional configuration layer on top of existing simulators, specifically tailored towards SDN-research. SEED allows to load saved configuration files and run simulations based on them. It separates the specification of the simulated network from the application to evaluate and the precise experiment specification. This allows to publish, review and reuse simulation scenarios independently from applications. By exposing a unified interface for different simulators, it aims to cover a variety of use cases and make the configurations more widely usable.

Keywords: Software-defined Networking, SDN, SDN-applications, networking, evaluation, reproduction, replication, repetition, simulation, toolchain

1 Introduction

Much effort in the research community of Software-defined Networking (SDN) is put into the evaluation of so-called SDN-applications which provide functionality such as routing or monitoring. Such an application sits on top of a logically centralized controller and alters network state through the controller's northbound API [KL15]. The evaluation of such applications is often done by using network simulators. The design of a meaningful simulative evaluation requires many choices on the utilized scenarios, the construction of those, their parametrization as well as the metrics to log. While all of these aspects have to be specified when configuring the simulation, many rarely change in different evaluations. However, since the specifications of experiments are rarely shared and more rarely reused, this time consuming and error-prone effort is often duplicated.

This paper therefore proposes the concept of a Shared Evaluation Environment for Software-defined Networking (called SEED). SEED is based on top of existing simulation environments and aims to ease the evaluation process of Software-defined Networking-applications as a whole from scenario design and specification over environment setup and simulation execution to collection and processing of results. It provides unified interfaces with which applications and scenarios are integrated into the simulation. Its design structure makes it

[1] Karlsruhe Institute of Technology, Institute of Telematics,
{addis.dittebrandt,michael.koenig2,felix.neumeister}@student.kit.edu

doi:10.18420/in2017_246

easier for researchers to share their created scenarios and applications. Other researchers can thus validate and reuse them in the evaluation of their applications, thus considerably reducing the overhead.

The contributions of this paper are:

- The identification of stakeholders towards SEED and their requirements.
- Insights gained through an exemplary paper reproduction (Shortcomings without using SEED).
- Results of a paper study, investigating the feasibility and requirements for SEED.
- The design of SEED.

2 Problem Statement and Gap Analysis

In order to make well-informed design choices, we conducted an analysis of current evaluation practices and their pitfalls. Section 2.1 identifies the stakeholders for SEED and their respective requirements towards it. This consideration is necessary to identify the aspects to look out for when examining current evaluation practices in sections 2.2 and 2.3. These sections outline the approach and findings of a paper reproduction and paper study respectively. At last, the resulting problem statement in relation to the findings in the previous sections will be given (sec. 2.4).

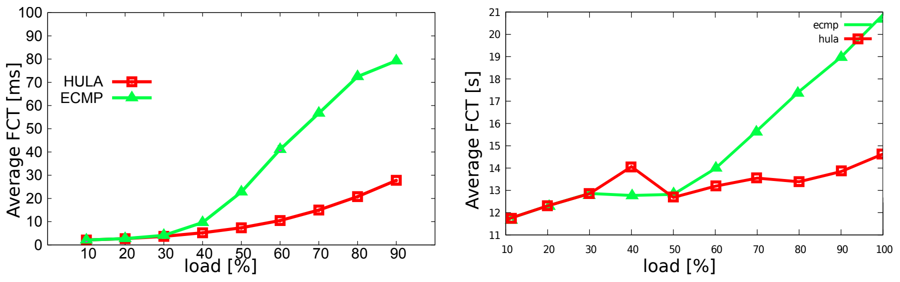
2.1 Stakeholders

We identified the following stakeholders that can use SEED.

Researcher As SEED aims to ease the evaluation of Software-defined Networking applications, the first stakeholder we identified is the researcher who has developed an application. When it comes to the evaluation of the developed application, the researcher has to set up the evaluation environment as well as a specific scenario. Furthermore, he needs to reimplement competing applications if they were not compatible with this environment. This poses a high overhead and it is, as will be shown, redundant and error prone. To ease this process, SEED needs to fulfill the following requirements: (1) SEED needs to enable the sharing of applications and scenarios. (2) These shared applications and scenarios must be openly accessible. The main problem that arises is how scenarios can be shared, e.g. the utilized format and interface. This problem does not arise with applications as they already utilize established northbound (e.g. Floodlight) or southbound (e.g. OpenFlow) interfaces.

Reviewer Artifact review (e.g. source code, evaluation environment) is becoming more and more important, as seen in recent discussion [Reb]. In order for reviewers to reproduce results, they have to obtain the artifacts from the researcher and set up the contained environment. This process is, as experience shows, often very complex due to a lack of standard structure and documentation. The same requirements as with the researcher hold true for the reviewer. Additional requirements are: (1) Given the application, the scenario and utilized parameters, replication of research results should be easy. (2) Scenarios must be easily exchangeable to enable evaluation with different scenarios.

2.2 Insights into Paper Reproductions



(a) Original Results [Ka16]

(b) Reproduced Results

Fig. 1: Initial results of paper reproduction (HULA)

In order to identify the problems associated with reproduction of evaluations, we reproduced the evaluations of papers in the context of Software-defined Networking. One of these papers was HULA [Ka16], a modern approach to scalable load-balancing.

The following difficulties to reproduce evaluation results subsequently lead us to the observation that reproduction and evaluation can be an elaborate and error-prone process, especially in regards to the details of traffic and topology creation, but also in terms of the configuration of the SDN-application itself.

To evaluate their algorithm the HULA authors used a data center topology, a fat-tree topology with two pods. For the traffic two workloads based on findings from [Al10] and [Gr09] were used by the HULA authors. The average flow completion times for three competing algorithms – namely HULA, ECMP (Equal-cost multi-path routing), and CONGA [Al14] – were compared for different levels of network-load.

For our reproduction we implemented both the HULA algorithm and ECMP as a ns-3 module and used the same fat-tree topology as the original evaluation. However, we used much lower link-speeds (4MBit/s, resp. 40MBit/s) compared to the original setup (40GBit/s) for our initial simulations, due long simulation times for higher bandwidths. To recreate the network-loads we tried to produce traffic with the same properties as used by the HULA authors. The traffic generated based on the few information provided in the HULA paper yielded in too different results. Therefore we additionally implemented a simpler load generator, which is based on bulk traffic senders, to create the necessary levels of load.

Compared to the original HULA evaluation both our HULA and ECMP implementation achieved qualitatively similar results in our reproduction setup for lower link-speeds (see curve progression of Fig.1 (a) and (b)). Here, our implementation of HULA attains the expected much lower flow completion times than ECMP, especially when used in cases with high network-load. For now, we could not reproduce the same, expected behavior for higher link-speeds. This could be due to different approaches to the traffic generation or due to deviating configuration of the HULA application itself (probe frequency, etc.). Therefore we are in contact with the authors about the details of the traffic generation.

Hence an uniform way to document and to pass on the necessary information in order to reproduce evaluation setup would simplify the reproduction process.

2.3 Paper Study

After we identified the above problems, we did a paper study to analyze how Software-defined Networking evaluation is done in a broader sense and to collect further requirements for SEED in order to simplify the evaluation process.

We chose 34 papers from various conferences in the field of Software-defined Networking (SOSR'16, SIGCOMM'16, CoNEXT'16, NSDI'16 and OSDI'16). The evaluation sections of the papers were of particular focus. The type of evaluation, utilized scenarios, controller software (for OpenFlow papers) and the benefit of SEED for those papers (if it were used for evaluation) were aspects of concern.

Most papers relied on experimental/testbed evaluation (75%), 40% on analysis and 25% on simulative evaluation (Many papers used more than one type of evaluation). As much as 40% utilized custom or very simple topologies. Otherwise we saw a higher prevalence of Campus (30%) and Datacenter (15%) scenarios. For papers that are set in the OpenFlow domain, we analyzed the utilized controller platforms as well. While many papers did utilize standard controller software such as Ryu (15%), Floodlight (15%) or OpenDayLight (8%), more than half of the papers employed custom controller software.

Therefore we gained the following key insights: (1) Datacenter and Campus scenarios should be provided with higher priority by SEED, as those were utilized the most after custom scenarios. (2) Support for a wide variety of controller platforms must be strongly considered, as there is no single platform used among the papers, but rather many entirely custom platforms. (3) SEED will act as an additional means of evaluation rather than being the main method. This is due to two factors: First, as SEED will be based on simulative evaluation and most papers having utilized the experimental approach for evaluation, having those migrate to the simulative domain can be regarded as too big of paradigm shift. Second, as most papers used either very simple or custom scenarios which are not necessarily representative, here, SEED could also provide additional means of evaluation on representative scenarios.

2.4 Conclusion

The insights gained from our reproduction efforts and the conducted paper study show that the evaluation and the reproduction of SDN-applications has reoccurring prerequisites: Selection and configuration of suitable topologies and traffic information, multiple parameterized runs of the SDN-application itself, setup of simulator or testbed as well as logging of needed measurement values and the following post-processing and interpretation of these measurements.

While the reproduction of an evaluation with all its necessities, such as fitting choices for traffic and topology as well as correct configuration of the SDN-applications seems to be error-prone and time consuming, coincidentally many research utilize the same kind of scenarios (i.e. with similar properties) to evaluate their applications.

Motivated by the similarities of these evaluation necessities we conclude that an environment that provides tools for these shared and common evaluation prerequisites and helps with the reoccurring tasks, could both improve and simplify the evaluation process.

3 Shared Evaluation Environment

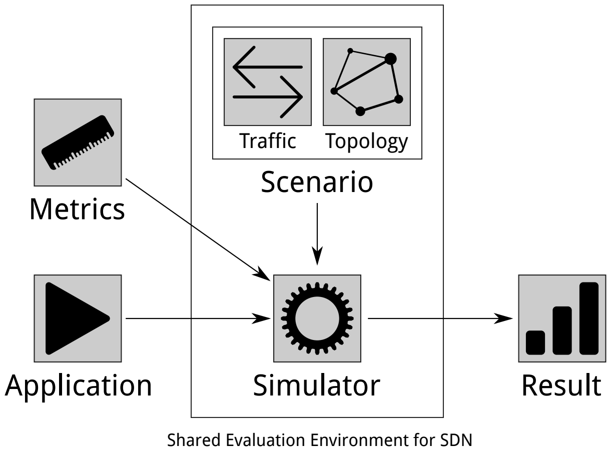


Fig. 2: Components of a Shared Evaluation Environment for SDN (SEED)

We envision to establish a shared evaluation environment, capable of covering as many as possible evaluation techniques in the context of Software-defined Networking. It should ease both the evaluation of SDN-applications and solutions as well as the review and reproduction of the evaluation. Our approach to achieve this goal is to structure and streamline the setup and configuration process of exiting simulators and make it easy for researchers to share their setup with reviewers and other researchers. Specifically SEED introduces an additional configuration layer on top of existing simulators. This should mainly reduce configuration overhead for both researchers and reviewers.

Since SEED is not part of the actual simulation, its focus does not lie on the correctness of simulations in general.

3.1 Components

SEED strives to make the sharing and reusing of configurations easier. This is achieved by separating the specification into three main parts: The application to evaluate and its setup (sec 3.1), the network and specifically the scenario to evaluate the application in (sec 3.1) and the specification of the experiment itself. This allows more comprehensive evaluation by making it possible to easily interchange the evaluation scenario. This interchangeability works both ways and also makes the evaluation of different applications in the same scenario easier, thereby encouraging the comparison of different approaches.

The separation additionally allows to bind the actual network configurations to independent, reusable bundles and publish them independently from applications, making it possible subject them to a review process. The application and its setup are the second building block in the configuration. It has to be specified independently from the evaluation scenario and able to stand for itself. These two building blocks are connected in the experiment specification. It gives, among others, details on events during the simulation, variables to log and additional configuration parameters for the two underlying building blocks. This specification should be the only component introducing dependencies between the scenarios and the applications.

Scenarios An integral part to evaluate SDN-applications are the scenarios and their network characteristics in which these applications are deployed. Depending on the nature of these network scenarios, an SDN-application might not be usable or behave differently. In order to evaluate the behavior of SDN-applications in different scenarios, the characteristics of the corresponding topologies and traffic need to be known and modeled.

A state-of-the-art study conducted by us beforehand has shown that modeling of traffic patterns and topologies with all the essential characteristics of a real network is often time consuming and that the usage of the available generators to create these have often non-obvious peculiarities such as special configuration options or that they are only suitable for niche cases ([BDP10], [Kr07]). Furthermore, the modeling of different contexts is itself still an ongoing research topic in terms of topology as well as traffic modeling (cf. the vastly different approaches to infer the shape of the Internet on AS-level). To use up-to-date and optimal fitting methods to create these evaluation requisites a researcher has to make extensive preparations and additionally each researcher has to redo these efforts each time so far on her own.

Therefore, we propose the idea of a collection of prefabricated combinations of topology data and suitable traffic information in so-called *scenario-bundles*. Each selected to match a specific scenario – e.g. a Data-Center Network with its fat-tree topology and corresponding traffic. For this purpose the lowest common denominator of similar scenarios from multiple evaluations could be used as an initial starting point to collect the necessary information for these *scenario-bundles*. These *scenario-bundles* in turn can then be utilized to recreate and compare them to existing research results in order to evaluate and improve their representativeness. The incorporation of feedback provided by a community using and sharing these bundles could improve the quality and accuracy of these bundles further.

The goal is to provide a "database" of these *scenario-bundles* to choose from. By that we want to release the individual researcher from the time consuming necessity to search for and select suitable and up-to-date tools to do this — hence the researcher can focus on improving and evaluating their actual SDN-applications.

Furthermore, such a collection of curated bundles enables the community to faster incorporate new ideas and current research findings. Subsequently replace outdated models with more representative ones more quickly. Therefore we want to develop these *scenario-bundles* publicly, along with the necessary tools to create them and the rest of SEED itself.

We explicitly invite the community as a whole to participate with feedback and ideas to establish such a crowdsourced "database" of *scenario-bundles* to be used as a shared source for various networking evaluations.

App integration Since fast setup and fast reproduction are key in the list of requirements for SEED, the setup and start of applications to evaluate have to be considered. Especially the installation of solutions has shown to be unnecessarily time consuming. Specifically the fact that the published software often lacks installation scripts and sometimes even lacks README files. SEED should help with this issue by making it easier to specify the setup and by defining a unified interface for installation, configuration, startup and tear down.

The way, the application interacts with the simulators has a considerable impact on the group of fitting simulators, their configuration and the setup and tear-down procedures. SEED combats this by requiring a dedicated configuration file for every software to test. This file allows to specify requirements for the environment, override its configuration and methods, specify the integration method (e.g. TAP-device or symbolic link) and link scripts for setup and tear-down, should the configuration options of SEED be insufficient.

The above introduced requirements not only benefit the goal of fast reproducibility: It is also beneficial for a researcher to have a clear starting point and streamlined configuration options. Our own experiences have shown that the early specification of installation, startup and tear-down functions significantly increases productivity over the long run.

Simulators As SEED is based in the simulative domain, it needs to be ensured that simulation in the context of Software-defined Networking, especially OpenFlow, is possible. An analysis of existing simulators was therefore conducted with regards to this capability, namely fs [So11], ns-3 [ns] and the OFSwitch13 extension for ns-3 [CGM16]. Those simulators were chosen as they all possess OpenFlow capablity. fs posed as the most promising simulator as it is a flow-level simulator which would enable evaluation in large scenarios. The analysis shows that it only provides support for a very basic OpenFlow version 1.0 featureset with features such as monitoring and modify-field actions missing. ns-3 is promising as it is a well-known packet-level simulator. Here, the analysis shows that ns-3 only provides an obsolete OpenFlow implementation. This is remedied by the OFSwitch13 extension. This extension provides functional and very complete OpenFlow v1.3 support, along with a switch model that incorporates flow table lookup delays. We therefore came to the conclusion that simulation in the context of Software-defined Networking is indeed possible with ns-3 and the OFSwitch13 extension being chosen as the simulator on which we base SEED. We still consider to incorporate other simulators or even the Mininet emulator [LHM10] due to its widespread use in this field.

Usability We plan to make this approach usable through a combination of software packets, documentation and guidelines for researchers on specifying their solutions. Here the use of suitable interfaces for the specification of scenarios, experiments and setup and startup procedures is key. All specifications will be done in XML-Files. This makes sharing and interchanging them much easier as oppose to using scripts. Additionally, each scenario or application should be contained in its own separate folder, making it easy to pack and distribute. The documentation and guidelines mentioned above should help make this separation easy to achieve without introducing major overhead.

3.2 Benefits to Stakeholders

The above identified stakeholders can profit in different ways from SEED. The researcher benefits from the clear structure and streamlined interfaces, since this reduces both the overhead for specification and the room for error. The capability to share and easily swap the network configuration also offers a number of advantages. Most importantly the possibility to reuse configurations from other researchers. This makes it easier to do a more comprehensive evaluation with a greater variety of scenarios. Lastly, it allows the independent sharing and reusing of parts of the specification, which makes it easier to get feedback from other researchers.

SEED also benefits reviewers. The structure allows reviewers to not only easily rerun shared experiments but also perform tests in other scenarios, making it easier to assess the general usability of a solution aside from the possibly hand-tuned scenarios, done by the researcher.

4 Related Work

SAFE [PMW12] and FNSS [SCP13] are existing tools that aim to ease evaluation in computer networks. SAFE streamlines the workflow of the ns-3 simulator by introducing a web interface where experiments can be dispatched among a set of worker machines. Results of the experiments are then collected into a central database with options to further process the collected data, e.g. by generating visualizations. FNSS is a toolchain which eases simulation setup and configuration by providing functionality in the form of a Python library to import and/or generate topology and traffic descriptions. FNSS provides support for multiple simulators (e.g. ns-3, OMNeT++ and Mininet) which consume these descriptions via adapters provided by FNSS. Both SAFE and FNSS lack explicit SDN support (the concept of SDN-applications and controller platforms). SAFE has no notion of a scenario format but instead relies on ns-3. This couples SAFE to the ns-3 simulator. While FNSS possesses a format for scenarios, traffic is specified through traffic matrices which we regard as insufficient. Furthermore, SAFE and FNSS are only concerned with parts of the evaluation process. While FNSS is primarily concerned with simulation setup and configuration, SAFE shows a bigger focus on execution of the simulation as well as collection and processing of results.

Such tools also exist in a broader scope. The Pantheon of Congestion Control [St] enables consistent evaluation and comparison of congestion control mechanisms by leveraging the common TCP protocol interface. Reproducing Network Research [Rea] collects research stories as well as replication instructions of papers in the branch of computer networks. It focuses solely on easing replication by providing clear documentation for this process and thus does not aim to ease evaluation at all. Collective Knowledge (cKnowledge) [FLP16] aims to promote and help researchers to create and share artifacts that are reusable and reproducible. Due to its broad scope, no unified interfaces exist that can be leveraged. Furthermore, cKnowledge is more tailored to systems programming.

5 Conclusion

We identified the need for a shared evaluation environment in the field of Software-defined Networking. Evaluation and reproduction are not trivial tasks which pose a high overhead and are potentially error-prone. We have identified researchers and reviewers as stakeholders for SEED and collected further requirements by conducting a paper study with a focus on how the evaluation of the respective papers was carried out and an exemplary paper reproduction on the HULA paper. We then presented the architecture of SEED as well as the concept of scenarios. We showed the feasibility of SEED with regards to simulators by analyzing the capabilities of them and selected ns-3 with the OFSwitch13 extension as a base for SEED. Preliminary results which already provide exchangeable applications and traffic generation indicate a promising direction for SEED.

Future work needs to be conducted on fully realizing SEED and provide first scenarios, namely a datacenter and campus scenario.

Acknowledgements

The conception and development of SEED is done in the context of a student project as part of the course “Research in Practice”, which is funded by “Bundesministerium für Bildung und Forschung” (BMBF) with the program “KIT-Lehre$^{\text{Forschung}}$” (FKZ: 01PL12004). It aims for an early incorporation of students into research. During a timespan of two semesters, students carry out all phases of a research project, namely the identification of the state-of-the-art, writing a project application, conducting the research project and finally writing the results in the form of a research paper.

Currently, the first half of the project is finished with the state-of-the-art and project application phases while the second half with conducting the project is underway.

Additionally we thank our supervisors R. Bauer, M. Flittner, M. Hock and M. Zitterbart for their feedback and encouragement. Further we want to thank the Institute of Telematics, KIT for the support and the equipment they provided.

Also we want to thank the anonymous reviewers for their comprehensive feedback.

References

[Al10] Alizadeh, Mohammad; Greenberg, Albert; Maltz, David a.; Padhye, Jitendra; Patel, Parveen; Prabhakar, Balaji; Sengupta, Sudipta; Sridharan, Murari: Data center TCP (DCTCP). ACM SIGCOMM Computer Communication Review, 40(4):63, 2010.

[Al14] Alizadeh, Mohammad; Edsall, Tom; Dharmapurikar, Sarang; Vaidyanathan, Ramanan; Chu, Kevin; Fingerhut, Andy; The, Vinh; Google, Lam; Matus, Francis; Pan, Rong; Yadav, Navindra; Microsoft, George Varghese: CONGA : Distributed Congestion-Aware Load Balancing for Datacenters. Sigcomm 2014, pp. 503–514, 2014.

[BDP10] Botta, Alessio; Dainotti, Alberto; Pescapé, Antonio: Do you trust your software-based traffic generator? IEEE Communications Magazine, 48(9):158–165, 2010.

[CGM16] Chaves, Luciano Jerez; Garcia, Islene Calciolari; Madeira, Edmundo Roberto Mauro: OFSwitch13. In: Proceedings of the Workshop on ns-3 - WNS3 '16. ACM Press, New York, New York, USA, pp. 33–40, 2016.

[FLP16] Fursin, Grigori; Lokhmotov, Anton; Plowman, Ed: Collective Knowledge: Towards R&D Sustainability. In: Proceedings of the 2016 Conference on Design, Automation & Test in Europe. DATE '16, EDA Consortium, San Jose, CA, USA, pp. 864–869, 2016.

[Gr09] Greenberg, Albert; Hamilton, James R; Jain, Navendu; Kandula, Srikanth; Kim, Changhoon; Lahiri, Parantap; Maltz, David a; Patel, Parveen; Sengupta, Sudipta: VL2: A Scalable and Flexible Data Center Network. ACM SIGCOMM Conference on Data Communication, pp. 51–62, 2009.

[Ka16] Katta, Naga; Hira, Mukesh; Kim, Changhoon; Sivaraman, Anirudh; Rexford, Jennifer: HULA: Scalable Load Balancing Using Programmable Data Planes. In: SOSR. 2016.

[KL15] Kim, Younggi; Lee, Younghee: Automatic Generation of Social Relationships between Internet of Things in Smart Home Using SDN-Based Home Cloud. In: Proceedings - IEEE 29th International Conference on Advanced Information Networking and Applications Workshops, WAINA 2015. IEEE, pp. 662–667, mar 2015.

[Kr07] Krishnamurthy, Vaishnavi; Faloutsos, Michalis; Chrobak, Marek; Cui, Jun Hong; Lao, Li; Percus, Allon G.: Sampling large Internet topologies for simulation purposes. Computer Networks, 51(15):4284–4302, 2007.

[LHM10] Lantz, Bob; Heller, Brandon; McKeown, Nick: A network in a laptop. In: Proceedings of the Ninth ACM SIGCOMM Workshop on Hot Topics in Networks - Hotnets '10. ACM Press, New York, New York, USA, pp. 1–6, 2010.

[ns] ns-3. `https://www.nsnam.org/`. (Accessed on 05/03/2017).

[PMW12] Perrone, L Felipe; Main, Christopher S; Ward, Bryan C: SAFE: Simulation automation framework for experiments. 978. IEEE, pp. 1–12, dec 2012.

[Rea] Reproducing Network Research | network systems experiments made accessible, runnable, and reproducible. https://reproducingnetworkresearch.wordpress.com/. (Accessed on 05/03/2017).

[Reb] Result and Artifact Review and Badging. https://www.acm.org/publications/policies/artifact-review-badging. (Accessed on 05/03/2017).

[SCP13] Saino, Lorenzo; Cocora, Cosmin; Pavlou, George: A Toolchain for Simplifying Network Simulation Setup. Proceedings of the 6th International ICST Conference on Simulation Tools and Techniques, pp. 82–91, 2013.

[So11] Sommers, Joel; Bowden, Rhys; Eriksson, Brian; Barford, Paul; Roughan, Matthew; Duffield, Nick: Efficient network-wide flow record generation. Proceedings - IEEE INFOCOM, pp. 2363–2371, 2011.

[St] StanfordSNR/pantheon: Pantheon of Congestion Control. https://github.com/StanfordSNR/pantheon. (Accessed on 05/03/2017).