Horse: towards an SDN traffic dynamics simulator for large scale networks

Eder Leão Fernandes[†]
e.leao@qmul.ac.uk

Gianni Antichi[‡]
gianni.antichi@cl.cam.ac.uk

Ignacio Castro[†]
i.castro@qmul.ac.uk

Steve Uhlig[†]
steve.uhlig@qmul.ac.uk

[†]Queen Mary, University of London [‡]University of Cambridge

CCS Concepts

•**Networks** → **Network simulations;** *Network experimentation; Network measurement;*

Keywords

Software Defined Networking; simulation

1. INTRODUCTION

The Software Defined Networking (SDN) paradigm can be successfully applied to the inter-domain ecosystem [1] to empower network fabrics with finer grained policies and traffic engineering capabilities. While there is potential to enhance Internet routing, introducing SDN at the inter-domain level might also lead to policy misconfiguration. As network fabrics are closely interrelated [3], configuration mistakes in one fabric can lead to unexpected shifts of traffic volumes in another one. Therefore, understanding the impact of new SDN dynamics is crucial for the design and management of robust inter-domain fabrics.

Figure 1 illustrates a network fabric comprised of edge and core switches. Such fabric has fine grained network policies and hence, packets are subject to different policy enforcements. Applications such as load balancing and blackholing, typical in legacy networks, should coexist in harmony with those enabled by SDN (i.e., specific peering and source routing). Composition of network applications has been studied in [4], however, ensuring no interference does not guarantee that packets flow as expected. Inconsistencies might occur even assuming completely independent policies. For instance, in Figure 1, a misconfigured load balancing policy can cause congestion in the core, a chosen source routing path might be inefficient, or a rate limiting policy can undermine the quality of a TCP transmission.

Simulators are a popular approach to verify network behavior and test applications before going in production. As SDN matured, a range of tools to reproduce networks were developed. Researchers and operators can use solutions such as Mininet[1] to study how fine grained policies may affect network traffic. Unfortunately, Mininet is not scalable [5]. This severely undermines the detailed analysis of use cases that involve large topologies and/or traffic loads. Furthermore, general purpose SDN simulators usually include the full implementation of the protocols stacks, e.g., the complete OpenFlow mechanics. Even though high fidelity is desirable, in specific situations, higher levels of abstraction may be necessary.

Finding the right level of abstraction to analyze SDN network policies is not trivial. Indeed, such an abstraction must be able to capture the interactions between the now decoupled control and data planes, and show the reaction of the controller to specific network events (e.g., a change in the path of a flow due to link congestion). At the same time, the degrees of freedom inherent to controller applications must be taken into account. The abstraction must allow a flexible logic for the SDN controller: suitable for testing but without explicitly dictating the simulated behavior. The trade-offs between flexibility and control are important, but finding the right balance is a key aspect in the design of a simulator.

Recognizing the need to evaluate policies over inter-domain SDN fabrics, this poster proposes *Horse*: a new simulator to foster SDN research and improve our understanding on the impact of the new use cases over the traffic flow. A simulation tool capable of efficiently reproducing large scale networks, high traffic loads, and policies, by abstracting the interactions between switches and controllers of the SDN network.

2. APPROACH

Figure 2 shows a high level architecture of the proposed simulator. The data plane of the simulator is based on three main building blocks: (1) Events, (2) Topology and (3) Traffic statistics, and network state. Events are a temporally ordered set of inputs for the

[1]http://mininet.org

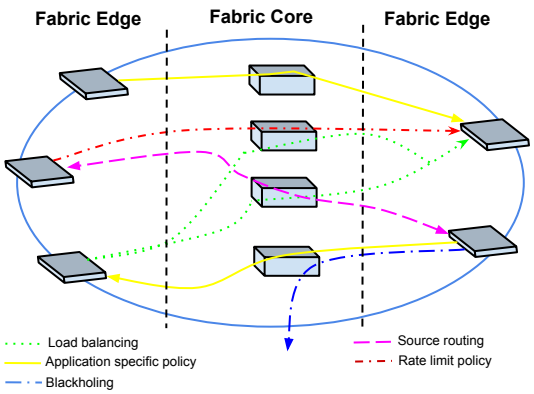

Figure 1: Traffic policies in an SDN network fabric. *Finer granularity in SDN enables complex policies across ISPs and IXPs networks.*

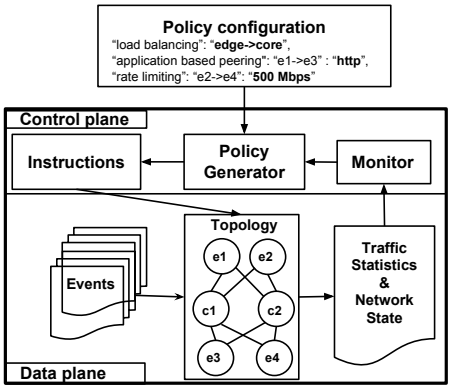

Figure 2: High-level architecture of the simulator.

topology (i.e, data traffic, link failure). To provide a scalable solution, we propose a flow-level data traffic representation, similar to fs-sdn [2], rather than a packet-level granularity. A data flow is an aggregate of packets with equal values of the header fields, but with different traffic rates. Data flows are treated as events that can be set in the form of a traffic matrix, or can be generated by the simulator itself. Traffic statistics and the state of the topology are updated after every event and exported to a control plane module. Although not reflecting the actual monitoring that would be performed inside a fabric, the monitoring primitives of the simulator will contemplate typical network measurements such as link bandwidth and SDN-enabled ones (i.e., OpenFlow counters). These measurements enable the creation of policies based on the current status of the network.

The control plane of the simulator is based on three building blocks: (1) Policy Generation, (2) Control Plane related instructions, and (3) Monitoring. The policy generator is a lightweight and modular controller that translates high level policies into OpenFlow control messages used as input for the fabric. Unlike other simulators, in order to reduce the state that needs to be kept, there are no real OpenFlow connections between the control and the data plane. While default policies will be driven by potential SDN use-cases[2], we target a flexible simulation infrastructure which enables the upgrading of the system with new policies (use cases). The policy generator will only make basic policy validation of policy composition.

We will evaluate the simulator by creating an SDN model based on the topology of one of the largest Internet Exchange Points (IXP). We will then assess the simulator using real data from the IXP itself, by replaying its behavior over time. Simulation time and accuracy will be evaluated under multiple configurations, from basic forwarding based on source and destination Media Access Control (MAC), to more complex combination of policies such as load-balancing and application-layer peering.

[2]https://www.h2020-endeavour.eu/sites/www.h2020-endeavour.eu/files/u54/D.4.1.pdf

3. ACKNOWLEDGMENTS

Research supported by the EU's Horizon 2020 research and innovation program (ENDEAVOUR project, grant agreement 644960).

4. REFERENCES

[1] A. Gupta, L. Vanbever, M. Shahbaz, S. P. Donovan, B. Schlinker, N. Feamster, J. Rexford, S. Shenker, R. Clark, and E. Katz-Bassett. Sdx: A software defined internet exchange. In *SIGCOMM*. ACM, 2014.

[2] M. Gupta, J. Sommers, and P. Barford. Fast, accurate simulation for sdn prototyping. In *HotSDN*. ACM, 2013.

[3] T. King. Traffic Volume Dependencies between IXPs. https://www.euro-ix.net/m/uploads/2015/10/26/e-TK-20151026-Euro-IX-Traffic_Volume_Dependencies_Between_IXPs-Neutral.pdf. [Online; accessed 15-Apr-2016].

[4] C. Monsanto, J. Reich, N. Foster, J. Rexford, and D. Walker. Composing software-defined networks. In *NSDI*. USENIX, 2013.

[5] D. Pediaditakis, C. Rotsos, and A. W. Moore. Faithful reproduction of network experiments. In *ANCS*. ACM/IEEE, 2014.