ez-Segway: decentralize for faster consistent updates

Keywords : Software-defined networking; Network update.

Thanh Dang Nguyen, Marco Chiesa, Marco Canini

Abstract – Recent research in Software-Defined Networking (SDN) introduced control plane mechanisms for updating the data plane while avoiding forwarding failures such as loops, black-holes or congestion. However, the update speed of these mechanisms suffers from latency between control and data plane as well as the overheads of centralized computation.

We propose ez-Segway, a mechanism that enables robust, decentralized network updates while still enforcing the desired endpoint policy and avoiding forwarding inconsistencies and congestion. In our architecture, the SDN controller only computes the intended network configuration and certain information needed to perform the update. This information is distributed to the switches, which use partial knowledge and direct message passing to efficiently schedule network update operations. Our evaluations via large scale simulations and a system prototype demonstrate that ez-Segway improves update performance by a factor of 2 and reduces message overhead by a factor of 3.

Many recent SDN systems have demonstrated the value of centrally controlling networks. We observe, like others before us [1, 3, 4, 6], that regardless of their goal, such systems operate by frequently updating the network configuration, either periodically or in reaction to events such as failures, load changes or policy changes. Updating network configuration is inherently challenging because it involves performing operations across different unsynchronized devices in multiple steps, each of which must be planned to avoid forwarding failures (e.g., loops, black-holes or congestion).

Previous works have proposed mechanisms to update the network while retaining certain consistency properties during the configuration changes. Reitblatt et al. [6] introduced the notion of "per-packet consistency", wherein every packet traversing the network must be processed by either the initial or final network configuration, even throughout the updated process. Their update procedure stamps packets with version numbers and uses a 2-phase commit protocol. Unfortunately, this approach is onerous in terms of memory overhead because it requires both the old and new forwarding entries to be present at switches' flowtables during the update. Moreover, 2-phase commit also delays the time at which the new configuration is used.

Consequently, others [3, 4] have explored different trade-offs between time, overhead, and update properties. And, per-packet consistency by itself does not admit reasoning about congestion. However, all these approaches have always commonly assumed that the SDN controller actively drives the update by (i) scheduling every step, (ii) sending rule to switches, and (iii) pausing to await ACK from switches. While, the switches just behave as remote passive nodes.

ez-Segway, decentralized network update. In contrast with prior methods, we investigate the prospect of delegating the responsibility of consistent updates to the switches. We first relax the per-packet consistency to a more general "*endpoint policy*" consistency, which specifies how packets should be processed by the network but not necessarily the exact path (e.g., a flow of packets must traverse a middlebox but the accurate path through the network is not relevant).

Based on this endpoint policy, we propose a distributed network update architecture wherein the controller is only responsible for computing the intended network configuration and pre-computing information needed by the switches to schedule network update operations. The actual update function is realized by the switches, which coordinate execution of an update for the entire network using the information received by the controller and direct message passing. This allows every switch to update its local forwarding rules as soon as the update dependencies are met (i.e., when a rule can only be installed after dependent rules are installed at other switches), without any need to coordinate with the controller. We show this approach leads to faster network updates, reduces the number of exchanged messages in the network, has low complexity for the scheduling computation, and constitutes the basis for a new class of algorithms that perform planned updates while reacting to accurate data plane measurements and conditions (e.g., link congestion [2], connectivity failures [5]).

Our model allows us to relax per-packet consistency into a more general endpoint consistency, to solve potential unavoidable link congestion by carefully splitting traffic aggregates volumes, and to leverage "flow segmentation", a novel technique to speed up the update of a single flow by parallelizing its update operations. Our results are as follows:

1. We design and implement ez-Segway, a system consisting of a reliable update scheduling mechanism that runs as software on switches, initially coordinated by a centralized SDN controller. Our algorithms enable decentralized network updates that preserve the endpoint policy while avoiding any forwarding failures and minimizing the risk of link congestion.
2. We assess our system by running a comprehensive set of emulations on various topologies and standard traffic patterns, which show that ez-Segway improves update time up to 2× and requires 3× less message overhead, and through extensive simulations, which also provide evidence of significant update time reductions for large scale networks up to 2×. Further, we validate feasibility by running our system prototype on a real SDN switch and present microbenchmarks that demonstrate low computational overheads.

References

[1] M. Canini, P. Kuznetsov, D. Levin, S. Schmid. "A Distributed and Robust SDN Control Plane for Transactional Network Updates". In INFOCOM, Apr 2015.

[2] D. Y. Huang, K. Yocum, and A. C. Snoeren. "High-fidelity switch models for software-defined network emulation". HotSDN 2013, August 16, 2013, pages 43-48, 2013.

[3] X. Jin, H. H. Liu, R. Gandhi, S. Kandula, R. Mahajan, M. Zhang, J. Rexford, and R. Wattenhofer. "Dynamic Scheduling of Network Updates". In SIGCOMM, 2014.

[4] H. H. Liu, X. Wu, M. Zhang, L. Yuan, R. Wattenhofer, and D. Maltz. "zUpdate: Updating Data Center Networks with Zero Loss". In SIGCOMM, 2013.

[5] J. Liu, A. Panda, A. Singla, B. Godfrey, M. Schapira, and S. Shenker. "Ensuring connectivity via data plane mechanisms". In NSDI 2013.

[6] M. Reitblatt, N. Foster, J. Rexford, C. Schlesinger, and D. Walker. "Abstractions for Network Update". In SIGCOMM, 2012.