Can the Production Network Be the Testbed?

Ashutosh Mittal (E-mail: amittal@kth.se)

I. Paper summary

THIS paper introduces *FlowVisor* which is a Network slicing tool. Network slicing refers to dividing network resources into different virtual networks that are isolated, independently managed to coexist over a shared physical network infrastructure. A common way of network slicing has been VLANs, but they have quite limited possibilities as compared to FlowVisor. It allows multiple network program to run safely and independently on the same production OpenFlow network. It is implemented as a protocol proxy that intercepts messages between open flow enabled switches and open flow controllers. Flowvisor layer is present between underlying physical hardware and software that controls it. It hosts multiple open flow controllers, one controller per slice to control that particular slice assigned to it. Each slice has a policy file which specifies its resource limits, flowspace, and controllers location in terms of IP and TCP port-pair.

FlowVisor slices resources in terms of bandwidth, topology, forward table entries, and device CPU. Each of these slices have control over a set of flows called flowspace. This technology also allows user to voluntarily opt-in or opt-out of any of the flowspaces.

FlowVisor does not require a one-to-one mapping between it's instances and physical switches. An instance of FlowVisor can slice multiple physical switches, and even re-slice an already sliced network.

The paper also evaluates FlowVisors scalability, performance, and isolation properties illustrating the efficiency and robustness of the design.

II. Significant Contributions

- *Allows real users to opt-in on a per-flow basis:* FlowVisor policy maps flows to slices. The mapping can be easily modified to try new services, and experimenters can have users bringing real traffic in the testbed. This allows expansion in terms of size and scale of testbeds as well as more rigorous testing of programming logic.
- *Operates on deployed networks:* Researchers generally work on individual testbeds which do not aptly mimic flow or size of actual networks. On the other hand, FlowVisor takes the testbed to the real network, enabling real world testing of programming logic. The authors of the paper have deployed it on their production campus network for at least 7 months before publishing the paper, consisting of 20+ users, 40+ network devices, a production traffic slice, and four standing experimental slices.
- *Enforces strong isolation between slices:* FlowVisor blocks and rewrites control messages as they cross the slicing layer. Actions of one slice are prevented from affecting another, allowing experiments to safely coexist with real production traffic. It has also allowed slicing in multiple ways: *Topology:* Each slice contains information about the network nodes and the connectivity between each of them. *Bandwidth:* Each slice has its own fraction of dedicated bandwidth on each link ensuring required throughput. *Device CPU:* Each slice has dedicated CPU resources as per the requirement and usage. *Forwarding Tables:* Each slice has a separate flow table ensuring no mix-ups and complete separation of routing rules of different programming logic.

III. Unresolved Issues

- Using openflow and software based control already adds some *latency*, and then on top of that we are using FlowVisor. I feel applicability of FlowVisor would be limited to research purposes and might not be useful for real time networks due to added latency.
- Another limitation is the inability to create *arbitrary topology* because a physical switch can appear only once in the topology. So, it can not be bigger than the existing physical network unless hardware loopbacks are allowed.
- Another issue is the *slice isolation violations in the device CPU*. Forwarding rule that could only be handled via the switchs slow path would push the CPU utilization to 100%, preventing slices from updating their forwarding rules.