previously installed FTEs. Nevertheless, deletion of the FTEs from the HwSw is not always possible. In fact, the sSwLogic checks the throughput (in terms of packets and bytes per second) contributed by the FTEs in the HwSw, deleting them only if their aggregated throughput could be afforded by the SwSw. In the other cases, sSw falls back to a direct installation of the new FTE in the HwSw.

When a *flow_mod* contains a FTE deletion request, the sSwLogic deletes the FTE either from the software switching layer or from both layers if the FTE was moved to hardware.

3. PROTOTYPE AND EVALUATION

We implemented a sSw prototype using a commercial hardware OpenFlow switch (HwSw) and an OpenvSwitch (OVS) instance running on a HP DL380G7 server equipped with an Intel Xeon L5640 (6 cores @ 2.26 GHz). The server is connected to the HwSw using both the switch's control port and two 1Gbit data plane ports. The sSwLogic, implemented as a user-level application, works as a proxy between the switches and an OpenFlow controller, which we implemented using POX on the same server. A second server, connected to two switch's ports, works as traffic generator and receiver. In all the tests we compare sSw performance with the performance of a high-end hardware OpenFlow switch.

***flow_mod* performance:** we measured the FTEs installation performance of the hardware switch when different FTEs priority patterns are used. Figure 2 shows the results. It is worth noticing that the FTE installation time increases almost exponentially with the number of already installed entries, when FTEs are installed in ascending priority order, demonstrating that TCAM entries reordering is happening. Instead, when FTEs have same priorities or they are installed in descending priority order, the installation time increases linearly with the number of entries in the forwarding table. Finally, FTE deletion time is much lower than the installation time and does not change in dependence of the FTEs' priorities (thus, we show a single curve for deletion in Fig. 2).

Our sSw prototype has two different performance behaviors. In the best case, it shows a constant FTE installation time, irrespective of the number of already installed FTEs and relative priorities, improving the hardware switch performance by 1 to 3 orders of magnitude. This is the behavior of sSw when the FTE that is going to be installed does not depend on any lower priority FTE in the HwSw. In case of dependencies, sSw deletes the dependent FTEs from HwSw in addition to installing the new FTE in SwSw. Figure 3 shows that the installation time grows linearly with the number of FTEs that have to be deleted. Also, Fig. 3 further shows that the deletion of several FTEs is (almost) always a faster operation than the addition of a single higher priority FTE to the HwSw. If the FTEs deletion is not possible, e.g., it would cause excessive load on the SwSw, sSw falls back to the hardware switch performance.

Reactive flow installation: in a second test we measure the installation delay of 10K network flows handled by a switch's port, using a reactive logic, i.e., a new flow generates an OpenFlow's *packet_in* message that is handled by the POX controller, which in turn installs a new FTE for such flow. We configure our traffic generator to timestamp the moment in which a flow's packet is sent and received. The flow installation delay is the time difference between these two timestamps for the first flow's packet. It is worth noticing that with this definition we include also the delay introduced by the POX controller, i.e., we look at the system as a black box. All the FTEs have same priority and define non-overlapping flows, which are generated at a rate of 1K flows per sec., over a 10 sec. period.

The measured flow installation delays are shown in Figure 4, where each flow is assigned with an id that reflects the generation order, i.e., first generated flow is assigned with id #1, the second

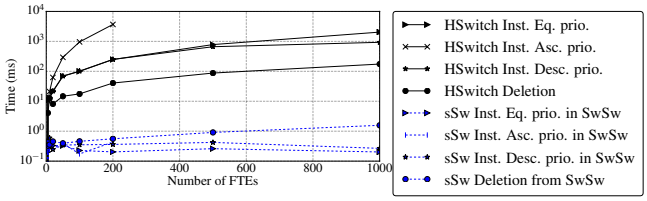

Figure 2: Cumulative execution time for forwarding table updates when installing/deleting a variable number of entries.

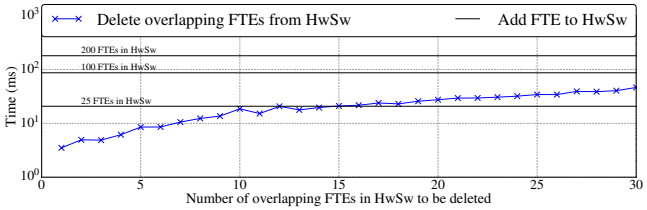

Figure 3: Time to delete a variable number of FTEs compared to the time to install a single FTE.

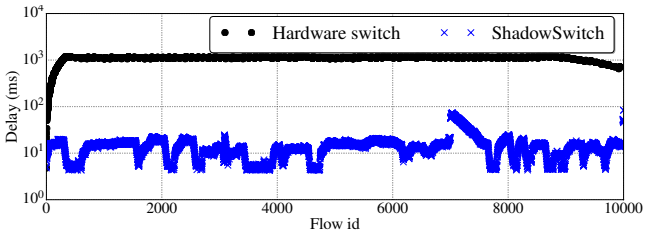

Figure 4: Flow installation delay (in log scale)

has id #2, and so on. The hardware switch drops about the 75% of the 10000 generated flows. The installation delay grows from the 5 ms experienced by flow #1 to about 1.1 s for flows with id bigger than 318. When using sSw no flows are dropped during the test, with an installation delay that is within 22 ms for the 95% of the flows. Figure 4 also shows that, when using sSw (lower line in figure), the flow installation delay experiences variations in the range 5 ms-20 ms. We believe these variations (and the higher delay peak for flows #7000 to #7500) are mainly introduced by our testbed, since we are running the POX controller, OVS and our user-space sSwLogic implementation on the same server.

4. FUTURE WORK

ShadowSwitch demonstrates that a fast software switching layer may help in reducing the flow table's entries installation time, however, the viability of the approach is dependent on the ratio between the software forwarding and hardware forwarding speeds, on the dependencies between flow entries, on the properties of the network flows in terms of packets and bytes per second. Our future work is to evaluate the system behavior when varying these parameters and to design smart algorithms that selects which entries should be moved first to the hardware flow table.

Acknowledgment

This work has been partly funded by the EU in the context of the "BEBA" project (Grant Agreement: 644122).

5. REFERENCES

[1] N. Katta, J. Rexford, and D. Walker. Infinite CacheFlow in software-defined networks. Proceedings of ACM SIGCOMM HotSDN, 2014.

[2] N. Sarrar, S. Uhlig, A. Feldmann, R. Sherwood, and X. Huang. Leveraging zipf's law for traffic offloading. *SIGCOMM Comput. Commun. Rev.*, 2012.