Paper Evaluation, Metron: NFV Service Chains at the True Speed of the Underlying Hardware

Jiani Jiang < jianij@kth.se>

1. Paper summary

Network Functions Virtualization (NFV) successfully changes the way that network services are deployed by supporting chains of network functions implemented on the top of hardware and brings the benefits like cost-saving for network service providers and flexibility for expanding service deployment. But the hidden problem is the physical cores allocation within a multi-core device during the packet processing. Limited ways of identifying cores are listed in this paper, including software switch dispatching method (E2), pipeline dispatching method with RSS (OpenNetVM, Flurries and NFP), and rule or hash-based hardware dispatching method (OpenBox, CoMb). Instead, a new system for NFV, Metron, was presented in this paper to perform a better solution with the results of higher throughput and lower latency during the packet handling. The key idea is ingenious hardware-based dispatching approach that can match the flow classes by OpenFlow rules with tags and then identify the core by matching the traffic to the server's NIC, so that every traffic class can match to a single core.

2. Top 3 contributions

a. The first paper to propose a new system for NFV services that can automatically and dynamically eliminate the unnecessary inter-core transfers to achieve packet processing at a high performance.
b. Make it possible to resolve network and server architecture mismatch issues adopting "run-to-completion" methods, by identifying traffic classes using SNF, tagging packets quickly and dispatching them throughout the service chain.
c. Make contribute to a new driver for programmable NICs and servers that can leverage popular management protocols to deal with the management of hardware from a commodity SDN controller and can easily merge together with new protocols.

3. Problems

It's hard to find problems, but I am confused about some statements in this paper:

a. In this design, the Metron controller will translate stateless operations into OpenFlow rules, and translate the remaining stateful operations into software instructions targeting the servers. My question is what operations is divided to stateless and what is stateful operation? How to decide which part is offload to the hardware? What's the trade-off between them?
b. The packet processing graph is a important role in this design which has a set of packet processing elements. But the paper doesn't mention that how and where dose it generates?