T4P4S[1]: A P4 compiler for a wide variety of targets

Presenter: Sándor Laki, Eötvös Loránd University, Budapest, Hungary (lakis@elte.hu)

In this talk, I will overview the recent advances of our multi-target P4 compiler called T4P4S (formerly called P4@ELTE) that currently supports various architectures like Intel x86 (using Intel DPDK), Freescale LS2085 (using Freescale's ODP-SDK) and TP-Link TL-WR1043ND (using OpenWRT and Linux UIO). Our aim with T4P4S was to develop a high-performance P4 compiler with flexible re-targetability. To this end, we have identified the essential components of the code, and split them into *hardware-dependent* and *hardware-independent* parts. Accordingly, the hardware dependent functionalities are defined by a Networking Hardware Abstraction Library (NetHAL) that has to be implemented for each target (DPDK, ODP SDK, Linux UIO), while the core compiler remains independent of the actual hardware, thus the core code and the actual target are interconnected through NetHAL. For both Intel and Freescale platforms our measurements show comparable results to standard example programs like L2, L3 forwarding examples, scaling well with the increasing number of cores. An early version of our compiler for Intel DPDK was presented at SIGCOMM 2016 as a demo, but this talk will shed light on the structure of the generated switch architecture, the design decisions we made during the development of the compiler and the latest performance measurements for both Intel, Freescale and OpenWRT targets.

To analyze the performance of T4P4S in realistic scenarios, four use cases from operational OpenFlow deployments were identified, and implemented in P4 and OpenFlow. The observed performance metrics provided by the compiled switch program with Intel NetHAL are compared to the ones resulted by OVS and PISCES, showing that the switch program generated by our compiler provides comparable performance to OVS with DPDK back-end and in many scenarios it even outperforms OVS.

The talk will also cover our ongoing and future activities including the support of P4-16, the next generation of P4 language whose community pre-release specification was published by the end of last year. There is an apparent similarity between the concept of architecture models and the NetHAL. Actually, the architectural model defines an interface, which can be implemented either in the compiler or in a library. In the official P4 approach, there shall be a separate compiler written for each target architecture, but in our method we relaxed this by using the compiler solely for synthesizing target-independent code and supplying the target-specific code as a library. Since for most of its abstractions NetHAL can be regarded as the implementation of the architecture model, we expect that our idea of component separation will equally well be usable in a compiler architecture tailored for P4-16. Finally, I will also show the first results on the efforts made towards the integration of T4P4S and existing SDN ecosystems (e.g. interoperation with OpenDayLight controller or OpenStack).

Project site: http://p4.elte.hu

Presenter's biography: Sandor Laki is an Assistant Professor at the Department of Information Systems, Eötvös Loránd University (ELTE), Budapest, Hungary. His research interests focus on active and passive network measurement techniques, traffic analytics, IP geolocation and algorithmic aspects of communication networks. Since 2015, he is a senior member of the Communication Networks Laboratory - a joint laboratory of Eötvös University and Ericsson Research. In the past years he has participated in various EC and EIT Digital projects, including FP7 OneLab2, FP7 NOVI, FP7 OpenLab, EIT Digital FITTING, EIT Digital Smart Ubiquitous Communication, Future Internet-PPP XIFI and FI-CORE, and H2020 MONROE. More information: http://lakis.web.elte.hu

[1] T4P4S (pronounce as TAPAS): Translator For P4 Switch programs